ARTIFICIAL INTELLIGENCE ENABLED AUTOMATIC TRAFFIC MONITORING SYSTEM

A Thesis presented to the faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfilment

of the Requirements for the Degree

Master of Science in Civil Engineering

---

by

VISHAL MANDAL

Dr. Yaw Adu-Gyamfi, Thesis Supervisor

DECEMBER 2019

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

ARTIFICIAL INTELLIGENCE ENABLED AUTOMATIC TRAFFIC MONITORING SYSTEM

Presented by Vishal Mandal,

A candidate for the degree of Master of Science,

And hereby certify that, in their opinion, it is worthy of acceptance.

____

Dr. Yaw Adu-Gyamfi

____

Dr. Carlos Sun

____

Dr. Prasad Calyam

ACKNOWLEDGEMENTS

Firstly, I would like to thank my advisor Prof. Yaw Adu-Gyamfi for always keeping his office doors open for me. No matter what time of the day it was, (at times even during 7 am morning weekends) he would find time for me. For this alone, he deserves the ‘Supervisor of the year’ award. Due to his amicable nature, I never hesitated to put forth my ideas to which he always supported and encouraged with his creative ingenuity. I consider myself fortunate to have had an opportunity of being supervised by a professor as kind, intelligent and as lively as him.

I would like to thank my parents, dad Jeebachh Mandal, mum Sushila Mandal and my younger brother Bishwas Mandal. They were my greatest source of support all these years. Their encouragement and suggestions helped me overcome many difficulties in life. It’s a privilege to have been blessed with such a loving and caring family.

I would like to thank the members of my thesis committee for their time, guidance and useful feedbacks which led to the completion of this MS thesis. I am grateful to my colleague Jin Peng for his help and support in coordinating the traffic anomaly study. Lastly, I would also like to thank all my friends for helping me through my master’s journey and making my stay in Columbia memorable.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The rapid advancement in the field of machine learning and high-performance computing have highly augmented the scope of video-based traffic monitoring systems. In this study, an automatic traffic monitoring system is proposed that deploys several state-of-the-art deep learning algorithms based on the nature of traffic operation. Taking advantage of a large database of annotated video surveillance data, deep learning-based models are trained to track congestion, detect traffic anomalies and tabulate vehicle counts. To monitor traffic queues, this study implements a Mask region-based convolutional neural network (Mask R-CNN) that predicts congestion using pixel-level segmentation masks on classified regions of interest. Similarly, the model was used to accurately extract traffic queue-related information from infrastructure mounted video cameras. The use of infrastructure-mounted CCTV cameras for traffic anomaly detection and verification is further explored. Initially, a convolutional neural network model based on you only look once (YOLO), a popular deep learning framework for object detection and classification is deployed. The following identification model, together with a multi-object tracking system (based on intersection over union – IOU) is used to search for and scrutinize various traffic scenes for possible anomalies. Likewise, several experiments were conducted to fine-tune the system's robustness in different environmental and traffic conditions. Some of the techniques such as bounding box suppression and adaptive thresholding were used to reduce false alarm rates and refine the robustness of the methodology developed. At each stage of our developments, a comparative analysis is conducted to evaluate the strengths and limitations of the proposed approach. Likewise, IOU tracker coupled with YOLO was used to automatically count the number of vehicles whose accuracy was later compared with a manual counting technique from CCTV video feeds. Overall, the proposed system is evaluated based on F1 and S3 performance metrics. The outcome of this study could be seamlessly integrated into traffic system such as smart traffic surveillance system, traffic volume estimation system, smart work zone management systems, etc.

CHAPTER 1 – INTRODUCTION

Monitoring traffic effectively has long been one of the most important efforts in transportation engineering. Till date, most traffic management centers (TMCs) rely on human operators to detect, track, alert and initiate response to different traffic incidents on our road networks. The overall procedure is time-consuming, expensive, not scalable for large road networks and overall not efficient as humans are prone to inaccuracies and subject to fatigue. In recent years, the demand for systems that can take advantage of current TMC infrastructure (CCTV cameras, sensors, probe vehicles, etc.) to automate traffic surveillance and incident management has been on the rise. The value of such systems, if they are able to considerably reduce the need for human interventions will be immeasurable.

Automated traffic surveillance systems using artificial intelligence have the capability to not only manage traffic well but also monitor and access current situations that can reduce the number of secondary road accidents. Additionally, ai-enabled systems should be capable of identifying each vehicle and track its characteristic to identify any dangerous driving conditions. They can also be used to detect stationary or stranded vehicles quickly and send alerts to agencies to respond to the incident. Stranded vehicles tend to impede traffic flow, hence early response or clearance could have significant impact on how fast traffic is restored to normalcy. Intelligent traffic monitoring systems are thus, an integral component of systems needed to alleviate the effects of traffic congestion and human factors.

As the world's population continues to rise, so does the number of vehicles and pedestrians using the roadways. The roads are over-utilized in a way that hampers the smooth mobility of vehicular traffic. There is always a need for an effective traffic surveillance system that monitors city's traffic and avoids any traffic bottleneck. One of the biggest issues causing traffic impedance and annoyance on the roads must be congestion. Therefore, congestion mitigation has been one of the top priorities set by almost all state Department of Transportation. Using an automatic traffic monitoring system capable of detecting congestion in real-time is not just able to alert traffic operators about the onset of congestion but also has the capability of reducing queue lengths, decreasing the number of accidents and overall helps lead a smoother flow of traffic.

In order to have a robust and fully versatile traffic monitoring system, anomalies such as congestion, stationary vehicles, traffic accidents, etc. need to be correctly detected. Automatic anomaly detection systems pose a challenging problem due to its necessity of functioning in real-time and under varied degrees of traffic and weather conditions. Another problem associated to detecting traffic anomalies is the quality of traffic videos obtained and the setbacks of occlusion that challenges vision-based system's detection capability. In this research, special emphasis has been laid on typical traffic scene anomaly detection and identifying the ways of palliating the effects of occlusion and poor video quality. Although there has been a large deployment of probe detectors and sensors for anomaly detection, the usage of vision-based systems can certainly prove more effective and less costly.

One of the major costs associated to monitoring current-day traffic at Traffic Management Centres (TMC) is to employ exceedingly larger personnel to monitor traffic by looking diligently at each CCTV cameras. To some extent, some machine learning techniques have been proven handy to monitor traffic. Although, it has not completely reached the stage at which traffic could be autonomously monitored without human intervention, but it can perform well with some level of automation under human intervention. This chapter gives a brief introduction about the existing challenges associated to traffic surveillance and presents the objectives of the current study.

1.1 Traffic Monitoring at TMC – Current State of the Art

Most Traffic Management Centres rely on human operators to monitor traffic. The process is highly labour intensive and subject to human errors. It becomes difficult for human operators to oversee traffic operations such as congestion, traffic volume, road accidents, etc. all at the same time on multiple cameras. Figure 1 shows how conventional traffic monitoring activity happen in most TMCs. Especially, when the operator works over 6 to 8-hour shift, the diligence might not always be high. Especially, when there are so many activities going on at the TMCs, from coordinating emergency response teams to reducing congestion bottlenecks [21], it is evident that the use of AI-based traffic monitoring would help relieve the work-load of human operators to some extent. Since the proposed approach, relies heavily on traffic cameras, it becomes a responsibility of TMCs to invest time in making sure that the cameras are working properly and if any difficulties are seen, overhaul the problems in the camera system in a timely manner.

TMCs are concerned with each traffic related activity happening on the roads from weather information, travel times, etc. to emergency and disaster management systems. Activities such as heavy traffic and congestion often impedes the flow of traffic, thereby causing delay in travel times. Therefore, one of the main activities in traffic monitoring is to detect congestion bottlenecks. Using an AI-enabled traffic monitoring system would help detect any congestion occurring on the road. The operators need not look into each of the CCTV cameras to spot congestion. This work could be automated in a way such that the proposed deep learning-based model is made to run in the background of video feeds coming out of every CCTV camera and automatically group them that have congestion. It is observed that on using deep learning algorithms trained on large datasets of traffic images, the model detected congestion with reasonably higher accuracies. Furthermore, the approach proposed in this study can detect congestion in different environmental conditions such as snow, rain, day-light, night-time, etc.

Figure 1. Human operator overseeing traffic operations [20]

Most Traffic Management Centres depend on infrastructure mounted CCTV cameras to monitor any traffic anomalies present on the road. The overall traffic monitoring activity including anomalies detection is highly labour intensive. The costs associated to having a human operator present at all times of the day and trying to verify traffic anomalies such as stopped vehicles, accidents, etc. can get very costly over time. Therefore, some level of automation is necessary to reduce costs and verify traffic anomalies in a timely manner. In this research, we aim at integrating a vision-based system for traffic scene anomaly detection. For this, a convolutional neural network is trained using a large database of annotated traffic images. The model so trained is then, used to spot and classify various anomalies on traffic scenes in real time.

1.2 Problem Statement

Traffic Surveillance in the United States and around the world are mainly done by human operators at the TMCs. In the last few years, there has been extensive research on machine learning-based traffic monitoring systems. The overall activity involving humans is highly time-consuming and has increased costs. In order to have a reliable traffic monitoring system, some level of automation is warranted that not just saves time but also helps reduce the overall cost of operating a TMC. Especially, activities like vehicle count and traffic density estimation cannot be conducted accurately by human operators and requires some artificial intelligence intervention. Similarly, when it comes to looking into traffic videos from multiple CCTV cameras, it becomes extremely difficult to analyse each traffic situation in real-time, especially for vehicle count and density estimation. Since the world is moving more so towards automation, most TMCs prefer deploying automated detection systems that can in fact, alleviate the work load of human operators and lead to timely and accurate detection. Therefore, for this research, a robust traffic monitoring system based on Artificial Intelligence technique is proposed so that some level of automation could be imposed in traffic monitoring and the overall work-load of human operators could be reduced. In this study, we deployed several state-of-the-art deep learning algorithms based on the nature of traffic operation so performed. Traditional algorithms often record lower accuracies and fails at capturing complex patterns in a traffic scene, hence we tested and deployed deep learning-based models trained on thousands of annotated database of traffic images. Thus, the system so proposed is capable of monitoring traffic including congestion detection, performing vehicle counts, detecting traffic anomalies and most importantly can scale to multiple traffic cameras.

1.3 Objectives of the study

- The main objective of this thesis is to develop a traffic surveillance system that leverages recent advances in machine learning to automate tasks such as vehicle detection, classification, counting, congestion and anomaly detection. The sub objectives of this study will include the following: Traffic Flow Estimation: Two main traffic flow variables will be estimated from video feed: traffic occupancy – the percentage of time a vehicle is in a location (pre-defined) on a video scene, traffic counts – aggregate of the average number of vehicles passing a section of a road by class (Trucks, Cars, Cyclists and Pedestrians).
- Traffic Scene Anomaly Detection: Leverage robust vehicle detection and tracking algorithms to detect anomalous traffic scenes such as non-recurring congestion, crashes, and stranded or stalled vehicles. A series of heuristics will be developed to ensure that the anomaly detection system can operate autonomously without the intervention of a human operator.
- Transferability and Scalability for Multiple Cameras: To enable large scale deployment of the system, models trained on a large variety of dataset will be developed. This will ensure that the performance of the system remains nearly constant across different cameras. A high-performance computing architecture will also be designed to ensure that the system can operate in real-time (or near real-time) even as the number of cameras or coverage increases.

CHAPTER 2 - LITERATURE REVIEW

Automated traffic monitoring systems are crucial for traffic management centres to save time and money, reduce congestion and increase safety on roadways. Present-day traffic management systems utilize a wide array of infrastructure-mounted CCTV cameras for remote monitoring of traffic conditions and verification of traffic incidents. Several studies have attempted developing vision-based systems to reduce reliance on human interventions for traffic surveillance through automation. However, these systems are often designed to address sub-components of traffic surveillance and therefore are not very useful for traffic management centres. A complete ai-enabled traffic surveillance system should be situationally aware, capable of performing multiple tasks simultaneously including: vehicle detection (i.e. both stationary and mobile), spotting congestion, detecting anomalous conditions, and overall monitoring the entire traffic scene. In this section, we broadly discuss some of the related articles focused on congestion detection, vehicle count and anomaly detection.

2.1 Traffic Congestion Detection

With the population rising steeply, the society is progressing towards urbanization and rapid industrialization. It is therefore, not surprising that traffic congestion mitigation has been one of the foremost directives of the US Department of Transportation. Ullman et al. in [22] hypothesizes that a congestion management system capable of detecting and speedily alerting response teams about the inception of congestion events can save both drivers and pedestrians' lives, diminish traffic queue lengths and overall improve movement through congestion bottlenecks. In order to detect and track road congestion events, some deep learning-based approaches are applied by several researchers. However, the first and foremost step in doing so would be to understand how traffic flows and what would be the short and long-term effects of congestion at certain road networks under varied conditions. Since, traffic monitoring happens on a larger scale, it is equally important to have a scalable, decentralized system that would provide a real-time feedback on traffic parameters for all road networks being surveyed. Fouladgar et al. in [23] proposed a decentralized deep learning-built system wherein, every node precisely predicted each of its congestion state based on their adjacent stations in real-time conditions. The main contributions of their approach were that it was easy to scale across different road networks and could be completely decentralized to predict the nature of traffic flows. For the first time, a neural network was used to model the flow of traffic and that's when congestion prediction came into existence using this approach in the early 90's [24]. In their work, the authors introduced a network that consisted of 1 input layer, 1 output layer and 1 hidden layer respectively. This design proved to achieve higher accuracy in predicting the nature of traffic flows and for estimating accurate travel times. However, its basic structure was inefficient for other traffic applications especially for the ones that contained a large dataset. With the advent of deep learning, basic single layered neural networks are not typically used for studying traffic flow and congestion parameters. Ma et al. in [25] proposed an entirely automated deep neural network-based model for analysing spatiotemporal traffic data. Their model first uses Convolutional Neural Network to learn the spatio-temporal features. Later, a recurrent neural network is trained by utilizing the output of their first step model that helps to categorize the complete sequence. Their model can have a possible application in studying traffic flows and predicting congestion.

Some recent studies have investigated using predicted traffic speeds for detecting congestion. When the predicted speed on a road segment is lower than the real-time speed by a certain margin, it is flagged for congestion. Wang et al. in [26] proposed a deep learning-based model that uses a recurrent convolution neural network structure to continuously predict traffic speeds. Using their model and integrating the spatio-temporal traffic information, they were able to identify the sources of congestion on city’s ring-roads. In general, most congestion detection systems obtain real-time traffic data from loop detectors or microwave radar sensors. These streams of data are then conceded through algorithms that perceives the start of congestion based on speed-volume and occupancy relationships and unceasingly tracks them as they grow and disperse. Skabardonis et al. [27], Bezuidenhout et al. [28] and Hourdos’ [29] research for example, proposed procedures which approximated congestion or queue lengths using combined loop detector data in 30-second interruptions. With frequent inadequacies of detector data, probe-based congestion detection substitutes have recently been considered. Dinh et al. [30], Wang et al. [31], Adu-Gyamfi et al. [32] and Cheng et al. [33] proposed end-of-queue and congestion detection systems through the input of high-resolution vehicle probe data through shockwave theory-based algorithms. From this, they testified very high detection precision on roadways with decent probe penetration rates.

Video or image-based congestion detection frameworks is also gaining popularity due to limitations of traditional sensors used for traffic flow data collection. Willis et al. [5] studied traffic queues classification using deep neural networks on traffic images by training a two-phase network using GoogLeNet and bespoke deep subnet and further used that for both image processing and congestion detection. They concluded their study showing that a deep-learned classifier system was able to detect traffic congestion with an accuracy of about 95%. Chakraborty et al. in [6] used traffic imagery and applied both DCNN and YOLO algorithms in different environmental conditions. Similarly, Morris et al. [7] proposed a transportable system for pulling out traffic queue parameters at signalized intersections from traffic video feeds. For this, they used image processing techniques such as clustering, background subtraction and segmentation to recognize vehicles and then finally, approximated queue lengths for various calibrated cameras at different locations of intersection. A similar approach was used by Hao et al. [8] to analyse traffic queue detection capacity through videos. The authors used the highest resemblance matching method and engrossed on a static background area window to remedy the deviations in real-time images and the lane region in setting. It is imperative to note that most automatic traffic monitoring system builds on the fact of how well vehicles are perceived. An unsupervised learning approach employed to categorize congested scenes using feature learning and density approximation is proposed in [16]. Likewise, a real-time computer vision system built on feature learning to track individual vehicles and monitor traffic is proposed in [4]. To quantity traffic parameters, a real-time computer vision system is presented in [17], that is capable of tracing vehicles under clogged conditions via a feature-based tracking algorithm. Similarly, Mandal et al. in [67] proposed a deep learning-built model that predicts traffic congestion using a vision-based system.

2.2 Vehicle Detection and Count

Vehicle detection and counting is an important component of any traffic surveillance system. With the higher processing speeds of GPUs, robust deep learning-based models can be trained to detect vehicles and tabulate counts in real time. Several traditional and non-traditional methods of vehicle detection and analysis have been proposed in recent years. Salvi in [1] proposed a method to automatically detect and track mobile vehicles in night-time or low-light traffic conditions. The author used a virtual loop-based approach to predict and tabulate the number of moving vehicles in different traffic scenarios. Wang et al. in [2] proposed an automatic vehicle counting technique using a DSP board and image processing technique. Mundhenk et al. in [34] created a dataset of overhead cars and used a deep neural network to classify, detect and count the number of cars. In order to detect and classify vehicles, they used a neural network called ResCeption. This network integrates residual learning with Inception-style layers that can detect and count the number of cars in a single look. Their approach is superior in getting accurate vehicle counts in comparison to the counts performed with localization or density estimation.

Most present-day counting methods could be broadly classified as detection instance counter [35-36] or density estimator [37-38]. Detection instance counters localize every car exclusively and then count the localization. However, this can have a problem such that the whole image is required to be scrutinized pixel by pixel to generate localization. Similarly, occlusions could create another hurdle as detector might merge overlapping objects. Density estimators work in an instinctive manner of trying to create an approximation of density for countable vehicles and then assimilate them over that density. They usually do not require too many samples of training data but mostly remain constrained on the same scene in which it is trained.

There has been a lot of interest in the areas of vehicle detection, recognition, and tracking of vehicles on different traffic scenes. Chiu et al. in [3] introduced an automated traffic monitoring system that implements an object segmentation algorithm capable of vehicle recognition, tracking and detection from traffic imagery. Their approach separated moving vehicles from stationary ones using a moving object segmentation technique that uses geometric features of vehicles to classify vehicle type. Their approach could be implemented on real-time traffic scenarios and is robust enough to detect vehicles that might otherwise remain undetected in cases of occlusion. Likewise, a real-time computer vision system for vehicle detection is proposed in [4] that uses a feature-based tracking algorithm. In their approach, the authors used a vehicle detection system that tracks the features of vehicles instead of tracking the entire vehicle every time. Thus, it eliminates the issue of partial occlusion and is more robust. The system can detect vehicles during both day-light and night time conditions with high degrees of precision. Zhang et al. in [9] applied deep learning on traffic video shots obtained from Unmanned Aerial vehicles to calculate the nature of traffic flows. In their approach, they used a Faster-RCNN to train vehicle detection model and capture vehicles from traffic videos. The accuracy of their proposed model was over 95% for different classes of vehicles. Likewise, the traffic flow statistic for light to heavy traffic condition was over 90%.

Advanced traffic monitoring system developed by Lin et al. [10] focuses on a stand-alone image tracking method for autonomously monitoring traffic. Their image tracking component can differentiate traffic information from detected positions of vehicles and is able to perform vehicle tracking in real-time. Zhou et al. in [11] proposed an example-based algorithm capable of detecting moving vehicles that would help automatically monitor traffic. The proposed approach is capable of effectively monitoring traffic under various environmental and lighting conditions. Salvi et al. [1] proposed a vehicle counting and detection system useful for traffic surveillance during low-light or night-time conditions. Bas et al used the adaptive background and kalman filtering approach to count vehicles from video-feeds [12]. In [13], a real-time vision system is proposed to automatically monitor traffic based on 2D spatio-temporal images. The proposed real-time vision system, VISATRAM can count the number of vehicles, estimate their speeds and can also classify them through 3D measurements.

Several recent studies on Convolutional Neural Networks (CNN) for vehicle detection and classification has generated higher performance capability over other algorithms. Bautista et al. in [14] used CNN to investigate its performance in correctly detecting and classifying vehicles that use low quality cameras. Ma et al. in [15] proposed a CNN that learns traffic as images, extracts traffic features and accurately predicts traffic speed. Their approach could be implemented to track vehicles and perform vehicular count as well. Likewise, Zhuang et al. in [39] proposed a statistical method that performs a correlation-based estimation to count city's vehicles through traffic cameras. For this, they introduced two techniques, the first one using a statistical machine learning approach that is based on gaussian models and the second one using the analytical deviation approach based on the origin-destination matrix pair.

2.3 Traffic Anomaly Detection

Early detection of anomalous traffic conditions such as crashes, stranded vehicles, etc. are necessary to reduce incident response times and reduce the likelihood for secondary crashes. The most important characteristic of traffic anomaly detection systems is their ability to perform both local behavioural analysis of each vehicle and a global flow analysis. Vehicle detection, tracking and information fusion are key components of typical anomaly detection systems. Kamijo et al. in [40] developed a vehicle tracking algorithm based on spatio-temporal Markov random field to detect traffic accident at intersections. Their model can track individual vehicle robustly without getting affected by occlusion and clutter effects, which remain a characteristic at most busy intersections. Traditionally, spot sensors were used primarily for incident detection systems [41], however, their scope tend to be rather trivial for anomaly detection systems.

Vision-based systems are utilized far and beyond mostly due to their superior event recognition capability. Information such as traffic jams, traffic violations, accidents, etc. could be easily extracted from vision-based systems. Rojas et al. in [42] and Zeng et al. in [43] proposed systems to detect vehicles on a highway using a static CCTV camera. Similarly, Lai et al. in [44] proposed a method to detect traffic violation at intersections. Their approach was put into practise in the streets of Hong Kong and was able to detect red light runners. Thajchayapong et al. proposed an anomaly detection algorithm that could be implemented in a distributed fashion to predict and classify traffic abnormalities in different traffic scenes [45]. Ikeda et al. in [46] employed image-processing technology to automatically detect abnormal traffic incidents. Their method was able to detect four different types of traffic abnormalities such as detecting stopped vehicles, slow-moving vehicles, fallen objects and the vehicles that endeavoured to change lanes consecutively. Michalopoulos et al. in [47] performed an auto-scope video-based prediction system to track incidents. In their approach, they demonstrated a method to track incidents up until 2 miles distance. An accident detection algorithm is proposed in [48] that utilizes the features of moving vehicles to automatically detect and record the picture of an accident before it occurred and afterwards. The picture so obtained was sent to TMC which could be used as a substitute for crash data and would be useful for further conflict analysis. The good thing about their approach is that it is fully automatic in detecting, recording and reporting any traffic accidents taking place at intersections, which could potentially improve the safety impact.

Recently, with the advances in the area of deep learning, there has been a lot of interest in utilizing various architectures of deep neural networks to detect anomalies. Wei et al. in [62] used Faster R-CNN to detect vehicles and introduced an unsupervised anomaly detection system based on background modelling. Liu et al. [63] proposed a region-aware deep model which extracted discriminative local features from a series of local regions of a vehicle. The authors in [64] put forward a generative adversarial network-based approach trained using normal data to detect the anomalies in images. It is reported that certain algorithms use multiple camera systems employing a system of video sensors to identify stationary vehicles. A method to combine vision-based tracking with Intersection over union would allow for an actual multi-object tracking technique that could be scaled over to detect traffic anomalies [51]. In [52], a machine learning technique to analyse traffic behaviour and detect the location of collision prone vehicles is proposed with high precision. Similarly, Yun et al. in [53] applied motion interaction interface to analyse anomalous traffic behaviour that can detect traffic accidents.

CHAPTER 3 - PROPOSED METHODOLOGY

The methodology adopted for implementing an automated traffic monitoring system is shown in Figure 2. The main components consist of first, a GPU-enabled backend (on premise) which is designed to ensure that very deep models can be trained quickly and implemented on a wide array of cameras in near real time. At the heart of the proposed AI-enabled system is the development and training of several deep convolutional neural network models that are capable of detecting and classifying different objects or segmenting a traffic scene into its constituent objects. Manually annotated Traffic images served as the primary source of dataset used for training these models. To enable the system to be situationally aware, different object tracking algorithms are implemented to generate trajectories for each detected object on the traffic scene at all times. The preceding steps are then combined to extract different traffic flow variables (e.g. Traffic volume and occupancy) and monitor different traffic conditions such as queueing, crashes and other traffic scene anomalies. The AI-enabled traffic monitoring system is capable of tracking different classes of vehicles, tabulating their count, spotting and detecting congestion and tracking traffic anomalies in real-time. The following sections will describe the details of each of the systems components including the algorithms deployed in this study.

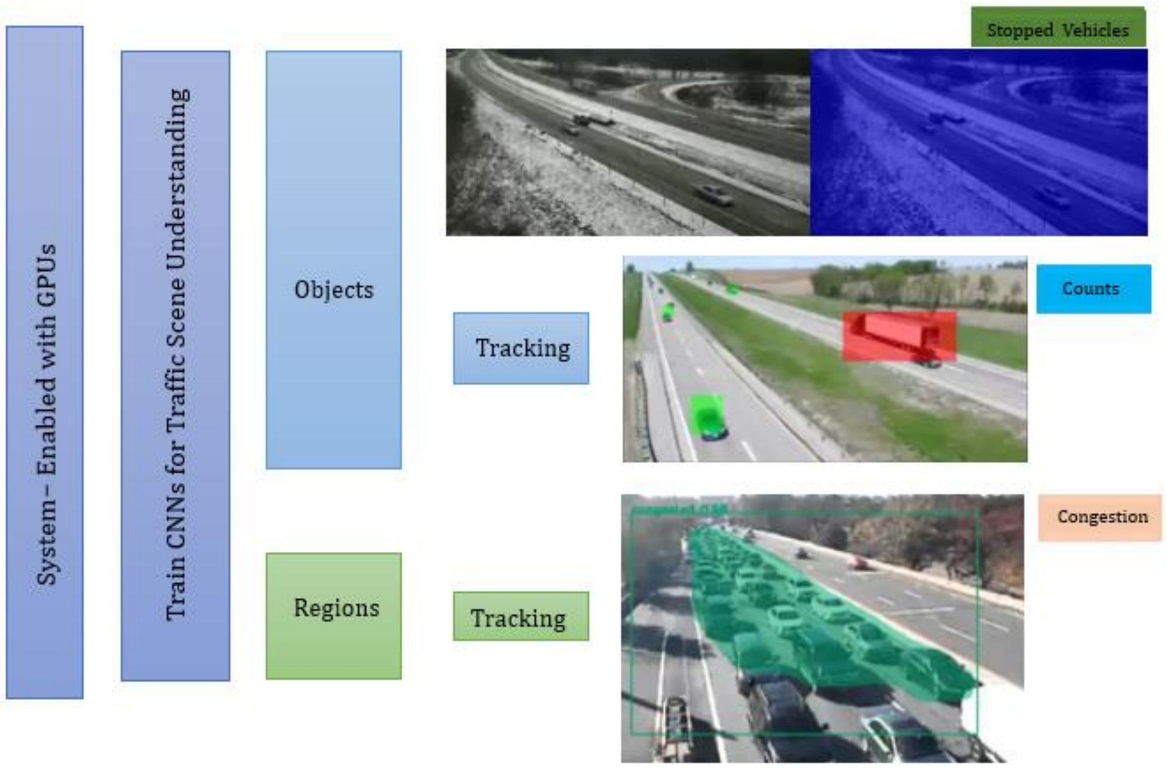

System – Enabled with GPUs
Train CNNs for Traffic Scene Understanding
Objects
Regions
Tracking
Tracking
Stopped Vehicles
Counts
Congestion

Figure 2. Visual Representation of the Proposed Research

3.1 Deep Learning

Deep learning is a branch of machine learning based on neural networks. It could be supervised, semi-supervised or unsupervised. Supervised learning is the one that learns labelled training data as an example and maps an output based on the level of learning. The training data is analysed, and an inferred function is produced that could be useful for generating newer examples. In ideal terms, a supervised algorithm can accurately determine class labels for unobserved instances. Similarly, semi-supervised learning is a class of machine learning that uses both labelled and unlabelled datasets for training purposes. However, the amounts of unlabelled training datasets are far more compared to the labelled set of training data. It has been observed that the learning accuracy is considerably improved when larger unlabelled set of data is used with smaller set of labelled data. Likewise, unsupervised learning is a machine learning technique that figures out unidentified outlines of dataset without any pre-existing labels. This type of learning mainly uses the principal component and cluster analysis. For Principal Component Analysis, an orthogonal transformation is used in order to alter a group of observations of correlated variables to linearly uncorrelated variables, also referred to as principal components. Similarly, for cluster analysis, the data that has not been labelled is grouped and is further clustered or categorized to evaluate common ground in datasets. This in fact, helps detect anomalous objects that do not appear familiar or do not fit into the group.

Basically, deep neural networks are in fact artificial neural networks with multiple hidden layers in between the input and output layers. Due to the presence of multiple layers, deep neural networks are capable of modelling complex non-linear relationships with complicated datasets which most similarly, performing single-layered shallow networks fail to do. The main advantage of working with deep learning algorithms is that they do not need structured or labelled data to perform prediction [18]. Deep neural networks rely on the hidden layers of artificial neural networks and learns through each layer hierarchically. The neurons of deep learning are classified into 3 different categories:

1. Input Layer
2. Hidden Layer
3. Output Layer

Figure 3 shows the graphical representation of deep neural networks. The input data is received through the input layer where it transfers the data to the first hidden layer. In Figure 3, we have altogether 4 different hidden layers. All the mathematical computation occurs inside the hidden layers. The data is computed in each first hidden layer and its output serves as the input of the second hidden layer and so on. It is therefore, important to understand what number of hidden layers and the number of neurons in each of these hidden layers would help get the best overall prediction. Similarly, after successive computations in each of the 4 hidden layers as shown in figure 3, the output layer returns the final output data, which we call the predicted value.

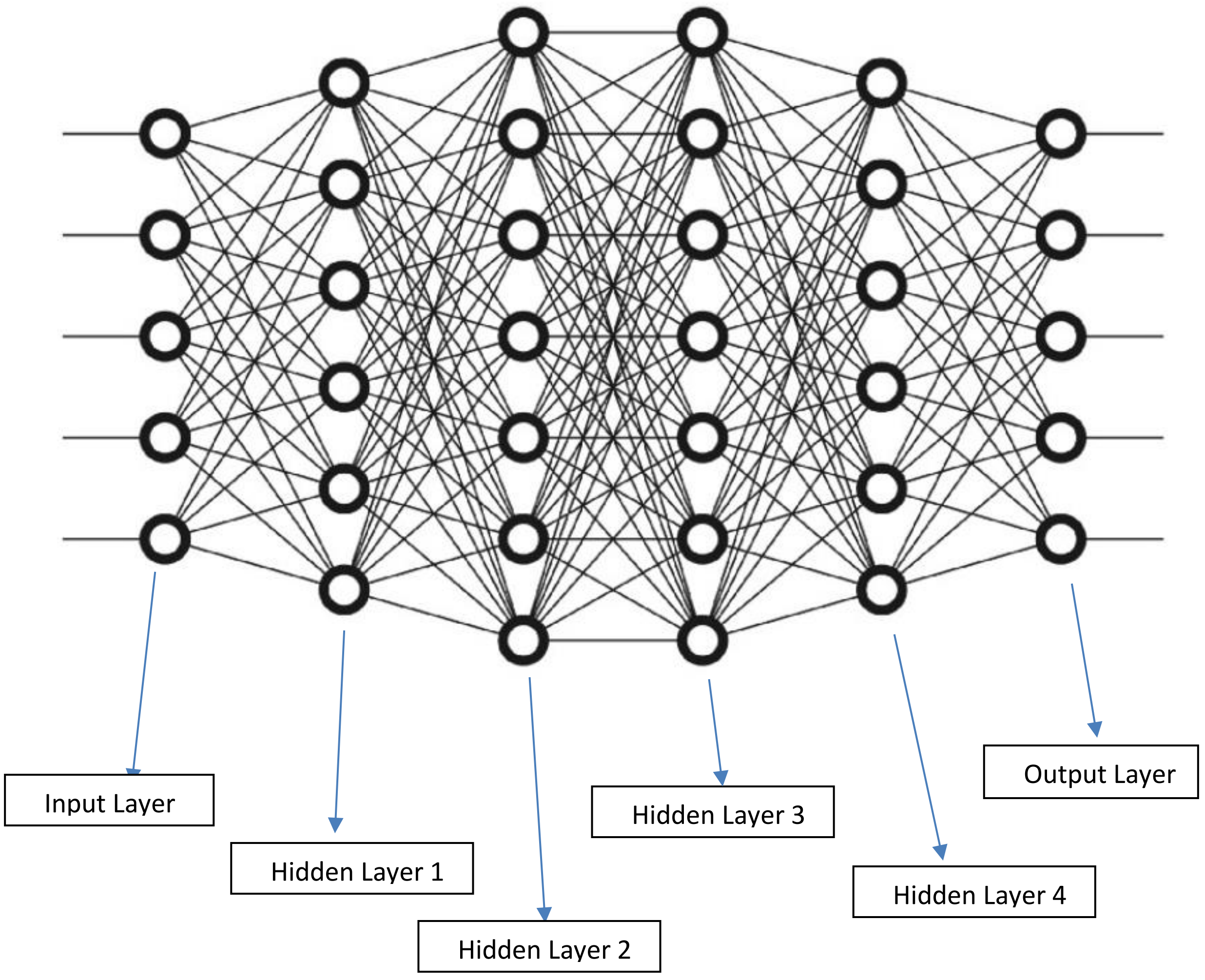


Figure 3. Deep Neural Network [19]

3.2 Deep Convolutional Neural Networks

In deep learning, ConvNet or a convolutional neural network is a section of deep neural networks applied mostly at performing image processing techniques. An image is passed through a deep convolutional neural network and assigned importance based on learnable weights and slight biases to certain features or items in the image that can differentiate one part of the object from another in same or dissimilar images. The pre-processing time required in deep convolutional neural networks is comparatively lower in comparison to other classification algorithms. Convolutional Neural Networks can purposefully seize both spatial and temporal dependencies throughout the image by applying numerous relevant filters. Especially, with the advent of GPU and improved algorithms, deep neural networks usage has surged several folds for analysing visual imagery. Some of the algorithms used in the current study are described as follows:

3.2.1 Mask R-CNN

Mask R-CNN abbreviated as Mask-region based Convolutional Neural Network is an extension to Faster R-CNN [56]. In addition to accomplishing tasks equivalent to Faster RCNN, Mask R-CNN supplements it by adding superior masks and sections the region of interest pixel-by-pixel. The model used in this study is based on Feature Pyramid Network (FPN) and is executed with resnet101 support. In this, ResNet101 served as the feature extractor for our model. It is imperative to state that the initial layers detected inferior characters such as edges and corners and the advanced layers could successfully distinguish higher-level features such as vehicles and traffic queues from the images. Similarly, on using FPN, it was observed that there was an improvement in the standard feature extraction pyramid by the introduction of another pyramid which took higher level features from the first pyramid and accordingly passed them over to subordinate layers. This enabled features at each level to obtain admittance to both higher and lower-level characters. In this study, we hard-coded the minimum detection confidence rate at 90% and ran it at 50 validation steps. Our model was run at 30th epoch where in every epoch had 100 iterative steps. An image-centric training approach was followed in which all the images were curtailed to the square’s shape.

The images were converted from 1024×1024px×3 (RGB) to a feature map of shape 32×32×2048 on passing through the backbone network. Each of our batch had a single image per GPU and every image had altogether 200 trained Region of Interests (ROIs). Using a learning rate of 0.001 and a batch size of 1, the model was trained on NVIDIA GTX 1080Ti GPU. A constant learning rate was used during the iteration. Likewise, a weight decay of 0.0001 and a learning momentum of 0.9 was used. The total training time for the model training on a sample dataset of 1,509 images, was approximately 3 hours. The main reason why the total training time appeared relatively shorter was because the number of images were not too many. It is important to note that with larger number of sample images in the training and validation dataset, the accuracy of results could be higher for masked detections. However, an issue with instance-level segmentation is that if there are larger number of training images, the accuracy improvement is not necessarily as high as expected. It is therefore, important to have effectively right number of images in the training sample to have acceptable detection accuracy. The framework for Mask-RCNN is shown in Figure 4.

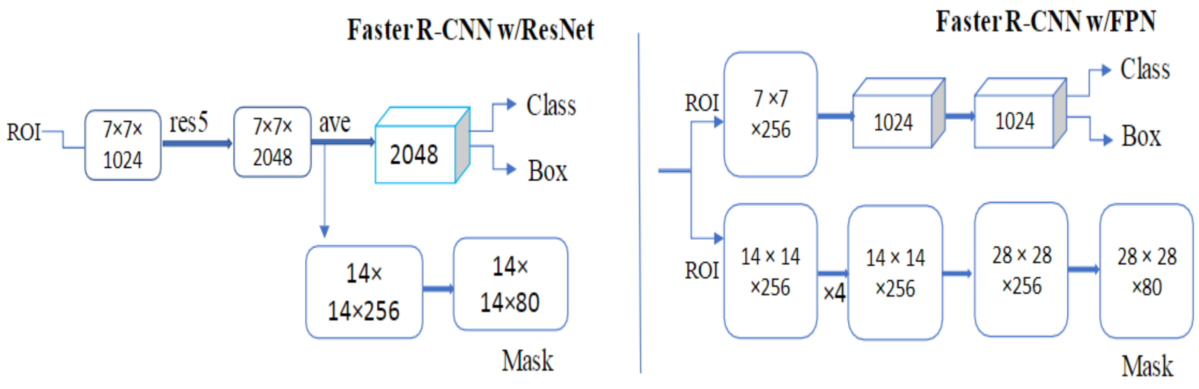


Figure 4. Mask-RCNN Framework

3.2.2 YOLO

You only look once (YOLO) is the state-of-the-art object detection algorithm. Unlike traditional object detection algorithms, YOLO investigates the image only once and detects any objects in it. In this research, we used YOLO to detect traffic queues, vehicle detection and perform vehicle counts. Most contemporary object detection algorithms repurpose CNN classifiers with an aim of performing detections. For example, to perform any object detection, these algorithms use a classifier for that object and test it at varied locations and scales in the test image. The good thing about using YOLO is that it reframes object detection that is, instead of looking at a single image 1,000 times to perform detection, it just looks at the image once and performs accurate object predictions. A singe CNN concurrently predicts multiple bounding boxes and class probabilities for those generated boxes. Due to this feature of YOLO, it is possible to easily implement it on diverse scenes. Table 1 shows the CNN architecture used by YOLO. The model uses standard layer types: A convolutional layer with a 3 × 3 kernel and a 2 × 2 kernel layer for max pooling. A 1 × 1 kernel is used for the last convolutional layer that helps resize data to the required shape of 13 × 13 × 125. To get the image apportioned, 13 × 13 structure size of the grid is used. We have altogether 35 channels for each grid cell which signifies bounding box data and class forecasts. For every grid cell, we get a total prediction of 5 bounding boxes and these boxes are designated by altogether 7 data elements in total. In order to build both traffic anomaly detection system and a congestion monitoring system on YOLO, the total training time was 22 hours for each of them separately. Like Mask-RCNN, YOLO was also trained on NVIDIA GTX 1080Ti GPU.

Layer	Kernel	Stride	Output Shape
Input			[416, 416, 3]
Convolution	3×3	1	[416, 416, 16]
Max Pooling	2×2	2	[208, 208, 16]
Convolution	3×3	1	[208, 208, 32]
Max Pooling	2×2	2	[104, 104, 32]
Convolution	3×3	1	[104, 104, 64]
Max Pooling	2×2	2	[52, 52, 64]
Convolution	3×3	1	[52, 52, 128]

Max Pooling	2×2	2	[26, 26, 128]
Convolution	3×3	1	[26, 26, 256]
Max Pooling	2×2	2	[13, 13, 256]
Convolution	3×3	1	[13, 13, 512]
Max Pooling	2×2	1	[13, 13, 512]
Convolution	3×3	1	[13, 13, 1024]
Convolution	3×3	1	[13, 13, 1024]
Convolution	1×1	1	[13, 13, 35]

Table 1. YOLO Model Architecture

3.2.3 Faster R-CNN

Faster R-CNN is a two-stage target detection algorithm. It is a third generation of R-CNN algorithm that associates altogether four basic iterative steps of target detection. In Faster-RCNN, a Region Proposal Network (RPN) shares a complete-image convolutional features alongside a detection network which enables cost free region proposals. Here, the RPN concurrently predicts object bounds and their corresponding score values at every position. Overall, end to end training of RPN produces high quality region proposals that is used by Faster-RCNN to perform object predictions. Faster-RCNN performs improved object detection compared to Fast-RCNN by replacing selective search method with RPN. The algorithm divides each image into multiple sections of reduced areas and then passes each area over a sequence of convolutional filters in order to excerpt rich feature descriptors which is then passed through a classifier. It is then that the classifier yields the probability of what certain areas of the image contains which kind of object. In order to obtain higher detection accuracy on traffic video-feeds, the model is twisted for 5 classes (pedestrian, cyclist, bus, truck and car) with the data obtained from traffic scenes. Altogether, 17,000 images were annotated and then used to train and validate the Faster-RCNN model. In order to effectively handle memory and speed requirement, NVIDIA GTX 1080Ti GPU was used. The overall training time was around 8 hours. It was observed that the resulting model was able to process video-feeds at the rate of 5 frames per second.

3.2.4 Comparison: Faster R-CNN, Mask R-CNN and YOLO

Both Faster R-CNN and Mask R-CNN are region-based object detectors whereas YOLO is a bounding box object detector where in a single convolutional network predicts the bounding boxes with class probabilities. In Faster R-CNN, instead of using selective search, a region proposal network is used which is a unified network for object detection. Faster-RCNN consists of two stages. The first stage is the deep fully convolutional network which proposes regions known as Region Proposal Network (RPN). An RPN is a fully convolutional network which predicts object bounds and objectness scores at every location. Here, the RPN serves as the attention of the unified network. In the second stage, the Fast R-CNN detector extracts features using RoIPool obtained from every candidate box and executes classification and bounding-box regression. Faster-RCNN takes image as an input which is then passed through the Feature network to generate feature maps. Similarly, Mask-RCNN is an extension to Faster R-CNN wherein an object mask prediction is run in parallel with the bounding box recognition. Compared to Faster R-CNN, Mask R-CNN is comparatively easier to train and only adds a small overhead to Faster R-CNN running at 5 frames per second. Faster R-CNN uses an RPN alongside Fast R-CNN for image detection. Basically, Mask R-CNN uses Faster R-CNN ground work for feature extraction and ROI proposals. After performing the ROI pooling for Faster R-CNN, an addition of 2 convolution layers is done to build the mask. Specifically, Mask R-CNN is a refinement of the ROI pooling. Likewise, Mask R-CNN performs the task of object classification based on ROIs just like Faster R-CNN.

Both Faster R-CNN and Mask R-CNN algorithms use regions to localize any object that may be present in an image. Neither of them looks into an image fully but only parts of the image that have higher probability value to contain an object. However, YOLO investigates an image fully and by deploying a single convolutional network predicts bounding boxes for objects in an image. Also, YOLO is much faster than both Faster R-CNN and Mask R-CNN and can process up to 45 frames per second. The speed of 45 frames per second could in fact be considered better than even real-time. The architecture of YOLO is more analogous to a fully convolutional neural network (FCNN) in contrast to RPN for both Faster R-CNN and Mask R-CNN. In YOLO, an image is passed through an FCNN and gives out an output prediction.

3.3 Traffic Congestion Monitoring

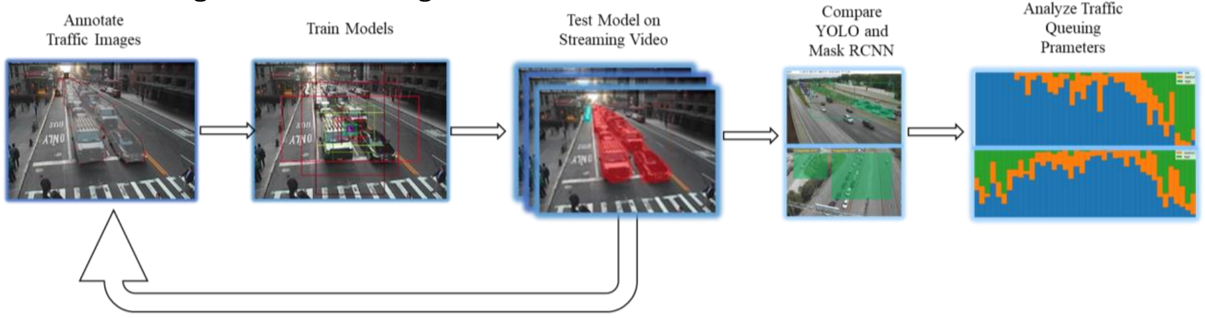

Annotate Traffic Images
Train Models
Test Model on Streaming Video
Compare YOLO and Mask RCNN
Analyze Traffic Queuing Prameters

Figure 5. Flowchart of the step-wise operations performed

The methodology adopted for implementing an automatic traffic monitoring system is shown in Figure 5. The first step was to perform annotations which was carried out using a VGG Image Annotator (See Section 4.3.1 for more details). Here, annotation consisted of a single class, i.e. only congested regions were selected. Then after, annotated images are used to train both Mask R-CNN (See Section 3.2.1 for more details) and YOLO (See Section 3.2.2 for more details) based deep learning models. The training time for Mask R-CNN and YOLO were approximately 3.5 and 22 hours respectively. Mask R-CNN was trained and tested from the models trained in between 30th epoch to 500th epoch. The best result yielded with the model trained on 30th epoch. Anything after that gave almost similar detection accuracy. Similarly, the trained models were tested on real-time traffic video feeds in order to evaluate their performance. In any case, when the model was not able to detect the congested scene, images were sampled from that scene for further annotation which was subsequently used for building a new model. When the model reached considerable levels, a comparative analysis was conducted in between Mask R-CNN and YOLO models.

Both Mask R-CNN and YOLO algorithms pose excellent object detection capabilities. Since, we aim at studying traffic parameters and tabulate the lengths of traffic queues, we used Mask R-CNN. The advantage of using Mask R-CNN algorithm is that it provides a pixel-wise segmentation approach and segments the region of interest pixel-by-pixel. Likewise, the proposed Mask R-CNN model can mine out queue related parameters from traffic videos. YOLO, on the other hand predicts traffic queues using a bounding-box approach. The most significant aspect of using a Mask R-CNN model for predicting traffic queues is that there is a pixel-wise segmentation of congested regions which makes the detections more precise. YOLO uses a bounding box approach to detect congestion where in the size of box covers exceedingly larger areas and the overall detection method may not seem as precise as the one performed with Mask R-CNN.

3.4 Traffic Anomaly Detection

The methodology so proposed for detecting traffic anomalies is shown in Figure 6. First and foremost, we deploy YOLO in order to train a deep convolutional neural network model for vehicle detection. Then after, the detections are tracked using Intersection over Union (IOU) and each vehicle trajectory is obtained from the traffic scenes. Tracking results are consequently used to outline certain travel directions (east, west, north or south), the kind of road being analysed, which could be either the intersection or freeway, and the predicted speed of tracked vehicles. The results of tracking are subsequently used to define distinct travel directions (east, west, north or south), the type of roadway being analysed (intersection or freeway), and the estimated speed of tracked vehicles. For certain kinds of roadway, if the speed of any vehicle drops down under a particular-threshold, then an anomaly is detected.

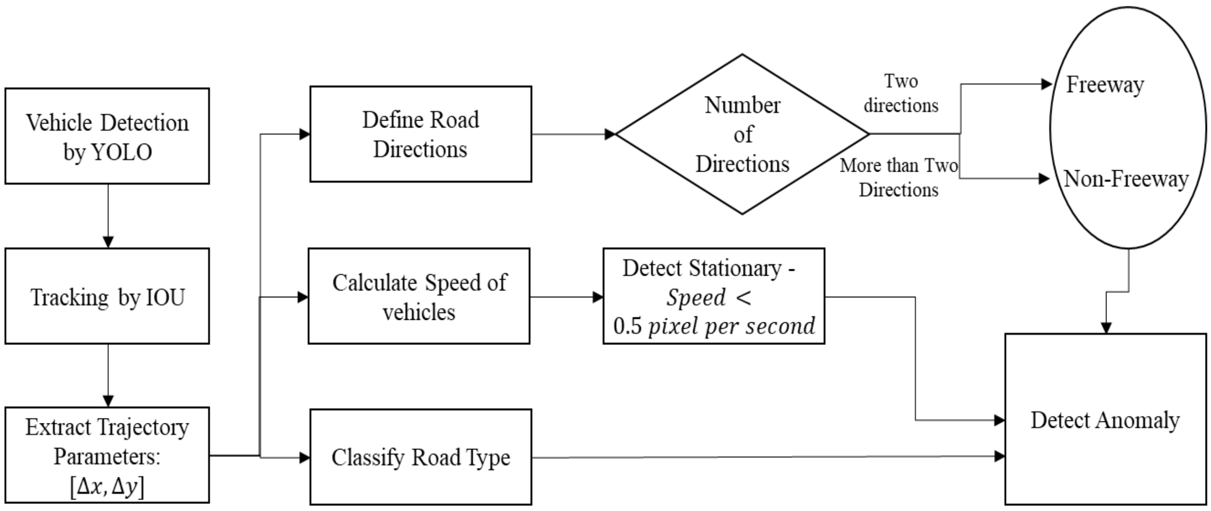

Vehicle Detection by YOLO
Tracking by IOU
Extract Trajectory Parameters: $[\Delta x, \Delta y]$
Define Road Directions
Calculate Speed of vehicles
Classify Road Type
Number of Directions
Two directions
More than Two Directions
Freeway
Non-Freeway
Detect Stationary - $Speed <$ 0.5 $pixel\ per\ second$
Detect Anomaly

Figure 6. Flowchart for traffic anomaly detection step-wise operations

The first step of the proposed method involves developing a model that is able of detecting and classifying all the vehicles in a traffic scene. In this study, we used Faster R-CNN and YOLO to perform object detection and classification. The purpose of implementing both these algorithms was to determine which one of them would turn out superior for object detection such that the one with higher accuracy and performance speed would be selected for further processing.

Upon successful object detection, tracking detected objects on traffic scenes allows the system to extract higher level features that are useful understanding video scenes. Once, the detector concludes locating the position of target in each frame, the tracker is accountable for obtaining the state of vehicles from the traffic videos. In order to associate detections to a unique track, multi-target tracking is used and based on the association results so obtained, the speed and state of each vehicle is estimated. Both locations based and feature based tracking is investigated for multi-object tracking.

3.5 Vehicle Detection and Count

Automatic analytics involving detection and counting of vehicles is important for many traffic applications. In this study, the vehicles were detected using YOLO and furthermore, tabulation of vehicles on both northbound and southbound directions is carried out using IOU. For each video, a line is drawn for north and south bound directions wherein the vehicles passing through each of these lines are counted and incremented accordingly. The figure 7 below shows the way these lines were drawn across the video. The red line across the traffic scene is drawn manually. Whenever, any vehicle passes through those lines, their count is recorded. Since, manual tabulation of the number of vehicles is a cumbersome task, the approach adopted in this study can significantly alleviate the work load of human operators. Most traffic counts in the TMCs happen through the adoption of sensors, therefore, the application of vision-based system could be a cheaper and a newer alternative to sensor-based techniques.

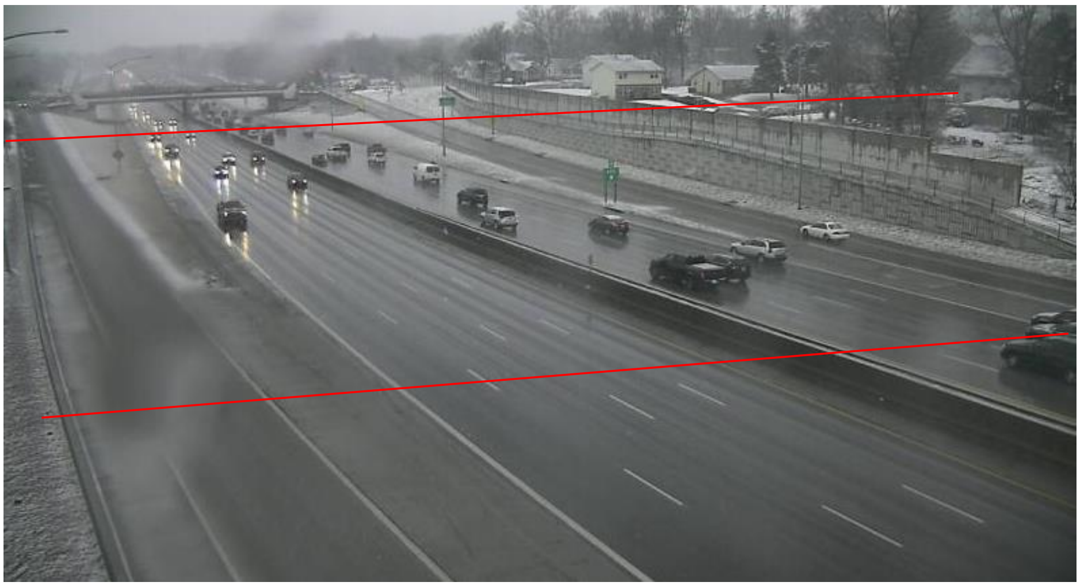


Figure 7. Lines Drawn Across Traffic Video Scene

CHAPTER 4 - DATA DESCRIPTION & RESULTS

4.1 DATA DESCRIPTION

Traffic Images served as the main source of dataset for this study. The images were obtained from Iowa 511, New York State DOT, RITIS and Iowa DOT Open Data. The over-all count of images obtained from Iowa 511, New York State DOT and RITIS were altogether 2,509. Out of them, around 1,509 images were used for training and 1,000 images were used for testing purpose. All these images were used to train and validate a Mask-RCNN model. Out of the 1,509 traffic images, 1,184 and 325 images were used for training and validation sets respectively. The datasets consisted of images taken at different times of the day in varied environmental conditions and contained congestion of all sort, from multiple sections of severely congested areas to the regions low on traffic. Training and testing samples of images consisted of the ones taken from different road types having diverse scenarios. Intersection, freeway and work-zone images were included in both training and testing samples. The images obtained from Iowa DOT Open Data were altogether 16,000. These images were used to train and validate deep learning-based models meant for vehicle detection, tracking and count. The traffic images consisted of both vehicles and pedestrians and were sub-divided into 5 different class types. These class types were pedestrian, cyclist, cars, buses, and trucks. For anomaly detection, traffic videos from ‘2019 NVIDIA AI City Challenge’ was used to test the effectiveness of the proposed model. Eventually, the model was assessed on 100 CCTV videos with different kinds of anomalies in erratic traffic and weather conditions [61]. An example of some of the traffic images obtained from Iowa 511, RITIS and New York State DOT under different environmental conditions and camera orientations are shown in Figure 8.

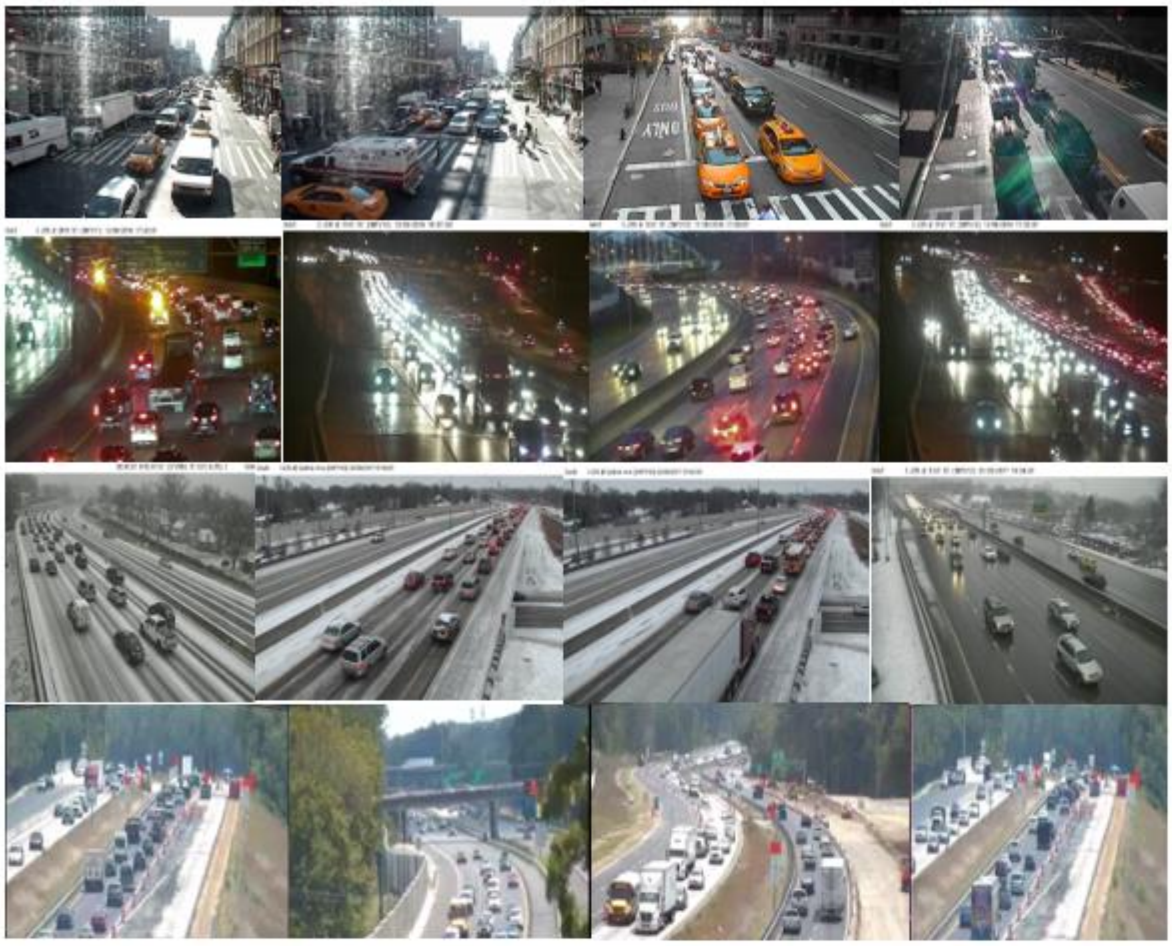


Figure 8. Traffic Congestion Images: 1st Row - Intersections during day, 2nd Row - Freeways at night, 3rd row - Freeways during snow, 4th Row- Work Zones

4.2 Model Training and Testing

Upon obtaining traffic images, the first step was to annotate them into different classes. Section 4.3 describes annotation in more detail. In this study, we manually annotated over 15,000 traffic images in VGG Image Annotator and OpenLabeling. An example of model training is illustrated in figure 9 where VGG Image Annotator is used to annotate congested regions from images. On manually annotating the images, the .json file which contains the annotated dataset is passed along with the original images to a convolutional neural network. With the help of a Graphics Processing Unit (GPU), the model training is carried out. Section 4.2.1 describes a GPU in more detail.

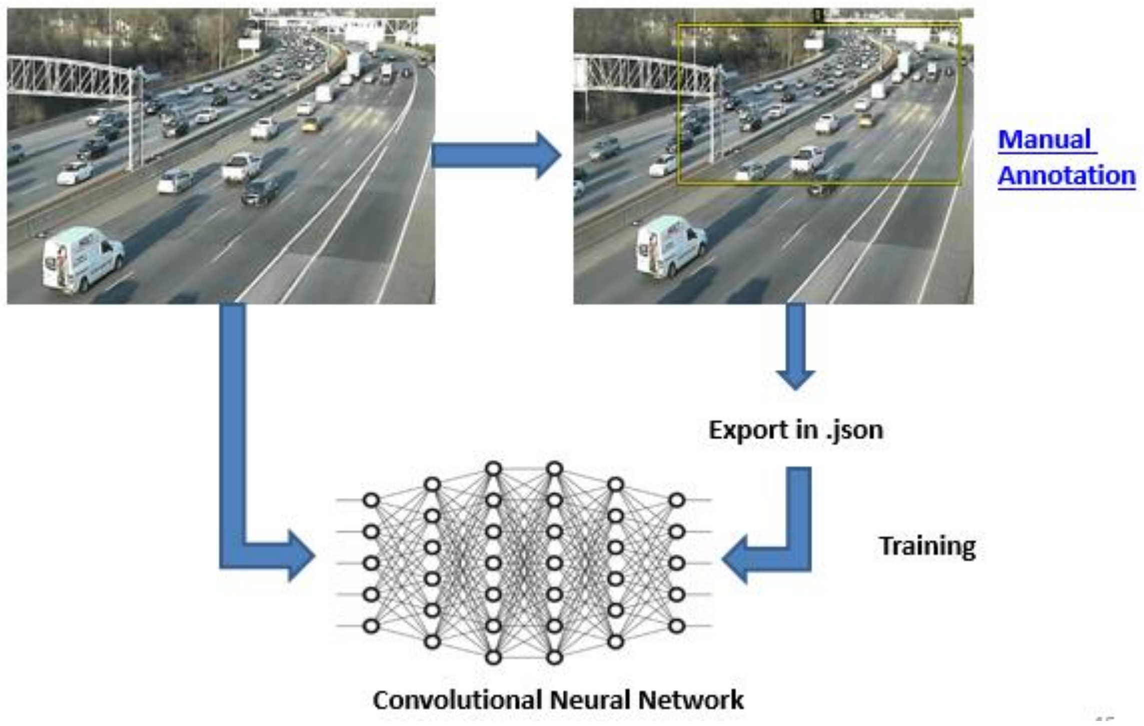


Figure 9. Training Convolutional Neural Network

4.2.1 Graphics Processing Unit (GPU)

A GPU is a programmable logic processor used to control memory in order to accelerate the creation of images in a frame buffer. They are typically specialized and useful for output to a display device. GPU processes images, and videos with exceedingly faster speeds compared to that of a Central Processing Unit (CPU). Although, CPUs are an essential component of a computer system, they are consistently moving from one task to another and requires calling up on information from hard drive repeatedly. However, GPUs break complex problems into individual tasks and work them out at a single time. This can turn out ideal for dealing with graphic and images and due its higher throughput can perform thousands of operations at once. On comparing the architecture, the GPU is composed of many cores that can handle multiple operations simultaneously whereas CPUs only have few cores. In the field of Artificial Intelligence, GPUs have become a key to deep learning technology. Since, deep learning requires large quantities of datasets through neural networks, a GPU is an essential aspect in training and performing tasks which would have otherwise, become impossible to do with a CPU alone. In this study, we used an NVIDIA GTX 1080Ti GPU as shown in figure 10 to train the deep neural networks.

Figure 10. NVIDIA GTX 1080Ti GPU

Several deep learning-based models were trained on this GPU and were evaluated to see their performance. Furthermore, fine-tuning was performed in order to obtain higher performance. Table 2 shows some of the training parameters for both Mask-RCNN and YOLO models.

Mask R-CNN Configuration	**YOLO Configuration**
STEPS_PER_EPOCH = 1000	Steps=40,000
EPOCH=30	Iterations= 120,000
BACKBONE = resnet101	backbone =Darknet-53
LEARNING_RATE = 0.001	learning_rate=0.001
LEARNING_MOMENTUM = 0.9	momentum=0.9
WEIGHT_DECAY = 0.0001	decay=0.0005

Table 2. Deep Neural Network Parameters

To test the deep learning-based models for congestion detection, a total of 1000 traffic images different from the training set of images were used where 500 of them were congested and the remaining were uncongested. Similarly, the proposed traffic scene anomaly detection algorithm was evaluated on 100 CCTV videos with different types of irregularities in varying traffic and weather conditions.

4.3 ANNOTATION

Annotation is the process of describing and defining spatial regions associated to an image typically by using standard region shapes such as rectangle, circle, polyline, freehand drawn sketch, etc. Annotation of digital images builds up on the fundamental processing stage for various research projects and industrial applications. Annotation is mostly carried out manually by humans. The spatial regions so defined manually are described using a textual metadata. In this research, we have used two different open-source annotation software tools named VGG Image Annotator and OpenLabeling respectively.

4.3.1 VGG Image Annotator

VGG image annotator is an open-source manual annotation tool that allows human annotators to describe regions in an image [54]. Although, rectangular shaped annotations are more common, VGG annotator allows to define freehand drawn sketch that could be aimed at training a deep neural network generating a pixel-level segmentation mask. The good thing about using a VGG Image Annotator is that it doesn’t require any installation of the software. It could be run on a web browser even offline without any installation or prior setup. It could be used for both image and video annotation. However, in our case we performed an image-based annotation. An example of the annotation performed using VGG Annotator is shown in Figure 11.

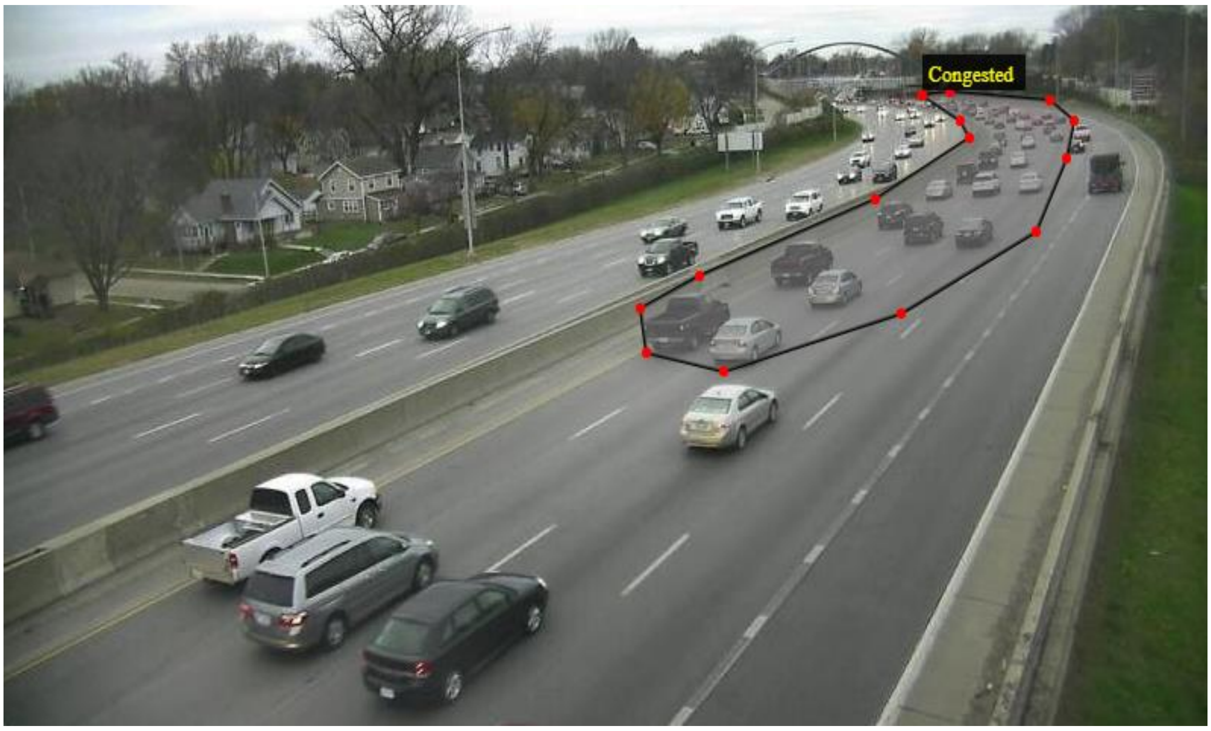


Figure 11. Annotation Performed Using VGG Image Annotator

4.2.2 OpenLabeling

OpenLabeling is an open-source image and video labelling tool that can label image in multiple annotation formats. It can be used with different deep learning features for computer vision applications. In our research, we used OpenLabeling tool to annotate different classes of vehicles that would later be trained on a deep neural network. We define the bounding boxes for those object instances that is further processed to train and validate deep learning-based models. An example of annotation performed using OpenLabeling is shown in Figure 12. Different classes of vehicles such as car and truck are shown in the figure below. An interesting feature of using this annotating tool is that there is an option to predict the next frames' label based on the annotation performed prior to that.

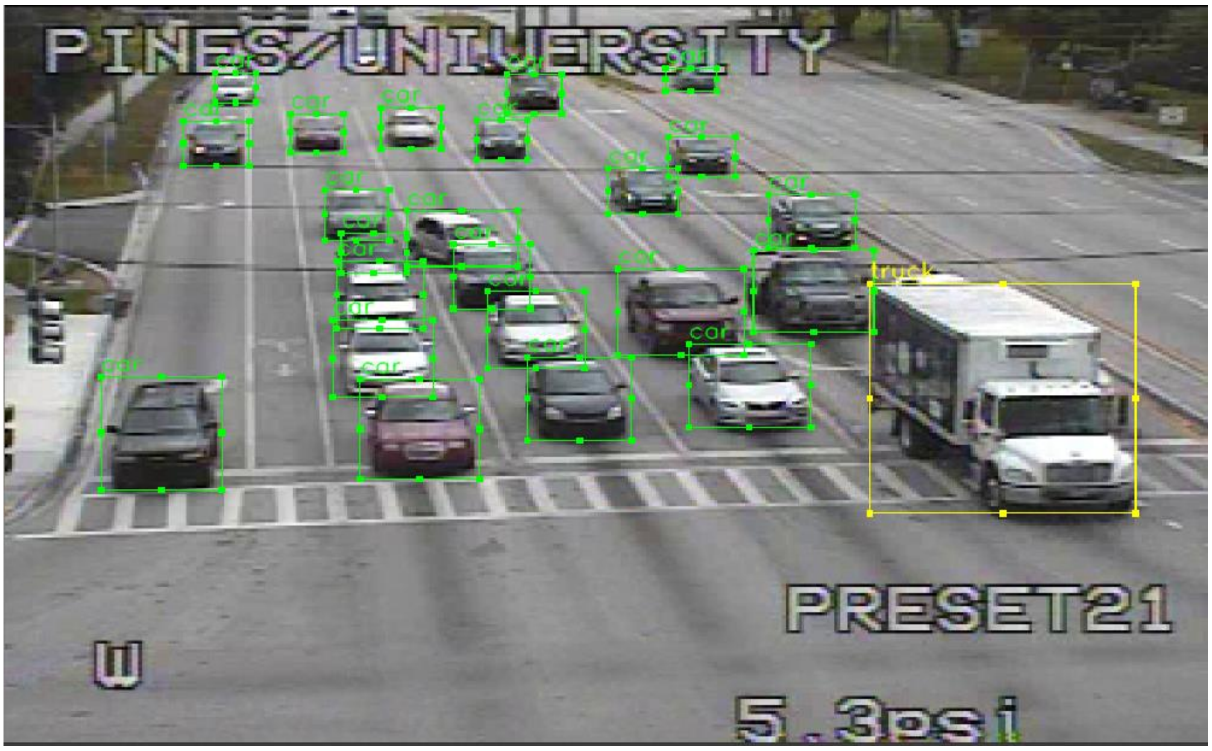


Figure 12. Annotation Performed Using OpenLabeling

After performing annotations for thousands of traffic surveillance images, we then train several deep learning based-models for congestion monitoring, vehicle counts, and for tracking road-side anomalies.

4.4 Results

In this section, we discuss the results of our proposed models. This section is further sub-divided into three categories. The first part discusses the results obtained for traffic queues detection and includes the study of various queuing parameters, the second part discusses the results obtained for traffic anomaly detection systems and the third part discusses automatic vehicle counting system.

4.4.1 Traffic Queue Monitoring

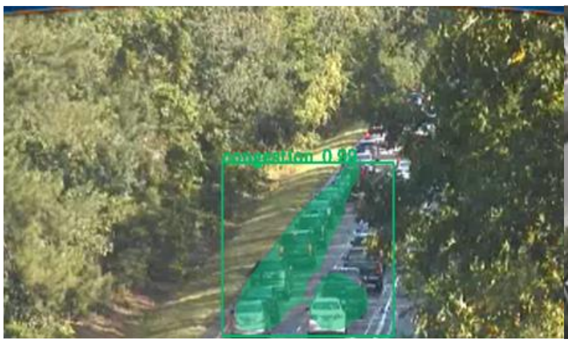
We first evaluate the performance of Mask R-CNN built traffic queues detection model on a set of 1,000 traffic surveillance images. Out of 1,000 images, 500 of them were congested and the remaining 500 were uncongested. The performance of Mask-RCNN was compared with the classical YOLO framework. Likewise, we discuss the results of a real-time application of Mask-RCNN for monitoring queues at freeways, intersections and work-zones. Standard performance metrics of precision recall and accuracy as shown in equations (i), (ii), (iii) respectively are used to evaluate both Mask-RCNN and YOLO models.

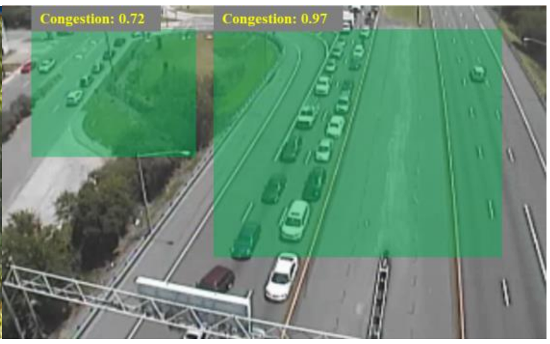
$$Precision = \frac{TP}{TP + FP} \qquad \text{(i)}$$

$$Recall = \frac{TP}{TP + FN} \qquad \text{(ii)}$$

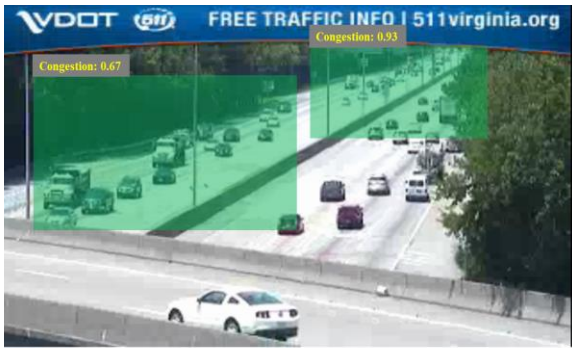
$$Accuracy = \frac{TP}{TP + FP + TN + FN} \qquad \text{(iii)}$$

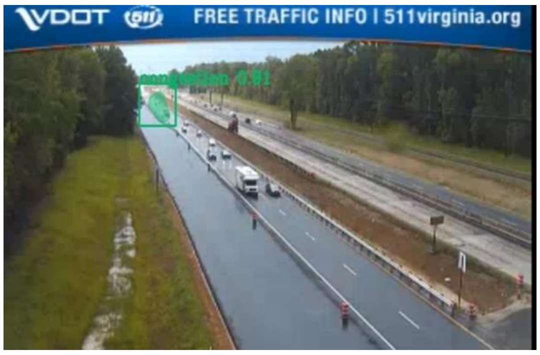
TP, TN, FN and FP are abbreviated as True Positive, True Negative, False Negative, and False Positive respectively. On testing, if a congested image is correctly categorized such that the predicted label is also congested, then that image is labelled as True Positive (TP). Similarly, if any image without congestion is categorized as uncongested, then that image is labelled as true negative (TN). For any instance, where the actual image is congested but then categorized as uncongested, the image is labelled as false negative (FN). Likewise, in cases where the image is uncongested but classified as congested, their labelling is done in the False positive (FP) category. The Figure 13 shows some of the correct detections and misclassifications obtained from Mask-RCNN and YOLO models:





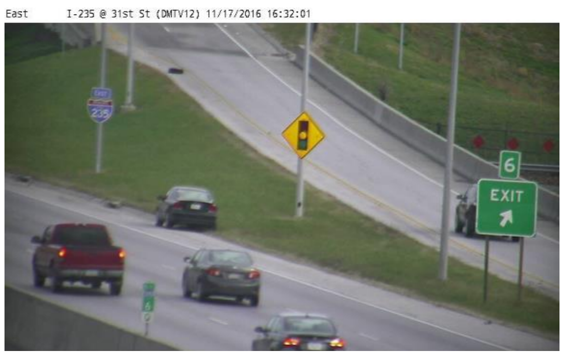
(a) (b)

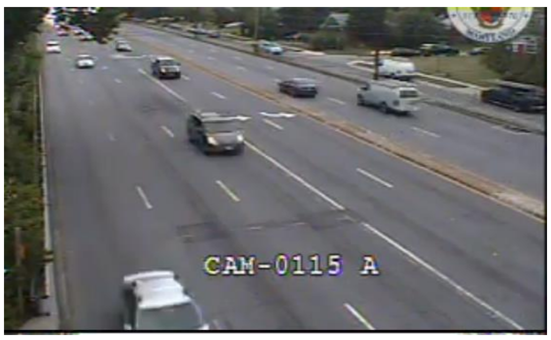




(c) (d)

(e) (f)





(g) **(h)**

Figure 13. Classification of predicted queues examples: True Positive-(a, b), False Positive-(c, d), False Negative-(e, f), True Negative-(g, h) obtained from YOLO and Mask R-CNN respectively

As seen from the figure above, 13(a) and 13(b) are accurate predictors for congestion and were classified as true positives. Figure 13(a) and 13(b) were the predictions made by Mask-RCNN and YOLO respectively. Mask-RCNN predicts congestion using a pixel-wise segmentation approach whereas YOLO predicts congestion using a bounding box. Similarly, Figures 13(c) and 13(d) were the misclassifications for non-congested images as congested. Hence, they are classified in the false positive category. Mask-RCNN erroneously predicted an uncongested image to a congested one because the image had a set of vehicles which appeared distant from the camera as seen in Figure 13(d). In case of YOLO, an incorrect prediction was made due to the presence of an overhead bridge which remain uncongested as seen from Figure 13(c). Illustration of false negatives are shown in Figure 13(e) and 13(f) where both Mask-RCNN and YOLO models failed at correctly detecting congested scenes. One of the reasons why YOLO was not able to detect congestion was because the group of vehicles appeared far away from the camera (See Figure 13e). Glaring effect as well as distant queues resulted in Mask-RCNN's incorrect classification as seen in Figure 13(f). Figure 13(g) and 13(h) accurately predicted uncongested images as false negatives as per the preliminary uncongested labelling.

The values for precision, recall and accuracy obtained for both Mask-RCNN and YOLO models are shown in Table 3. YOLO attained higher precision and accuracy values, but then again, a lower recall value as compared to Mask-RCNN. It is apparent that the general performance of Mask-RCNN is like that of YOLO. The purpose of using Mask-RCNN is that it provides a pixel-level segmentation of congested regions that makes queues detection much more precise. Although, queues detection by YOLO is accurate, its ability of predicting queues using a bounding box approach makes it non-viable for studying traffic queue parameters that could otherwise be done by Mask-RCNN. It is mainly due to the precise detection capability of Mask-RCNN as it covers only selected regions of interest that make it feasible for this study. Thus, in context of traffic queues detection and analysis of queue related parameters, Mask-RCNN outclasses YOLO by providing an accurate congestion measure.

Model	Precision (%)	Recall (%)	Accuracy (%)
Mask-RCNN	92.8	95.6	90.5
YOLO	95.5	94.8	93.7

Table 3. Precision, Recall and Accuracy Values Obtained from Mask-RCNN and YOLO

4.4.1.1 Case Study: Extracting Queue Parameters

In this section, we perform a case study where the Mask RCNN model developed is executed in real time for traffic congestion monitoring at an intersection, on a freeway and construction work zone. It is imperative to note that the distortions in video camera perspective often make it challenging to excerpt traffic queue parameters from video-scenes. A characteristic course around this is to adjust the camera to a certain height, viewing angle, zoom level, etc. Although this might be effective but is not scalable. Another substitute to this approach would be to directly use image pixel values to characterize queue parameters. On using this approach, queue information from one site location cannot be compared to a different site since the camera geometric configurations may vary. In the steps explained below we develop a simple, standardized calibration free technique for extracting queue length parameters from traffic video feeds. This technique is scalable and is useful at comparing queuing levels at multiple locations.

Step 1: Extract queue regions from traffic video-feeds with Mask RCNN.

Step 2: Calculate the pixel length of each detected queue mask.

Step 3: Accumulate length over time (minimum duration is 1 week).

Step 4: Use adaptive thresholding (Figure 14) to bin queue lengths into different severity levels: low, medium and high.

Step 5: Generate heat map of queuing levels and finally, compare.

```
Steps shown for Adaptive Thresholding
Initialize: L, M, H
Input: PL – pixel lengths
for each location do
    for each $[\mathrm{day}, \mathrm{hour}, \mathrm{minute}]$ in $[30\ \mathrm{days}, 24\ \mathrm{hours}, 60\ \mathrm{minutes}]$ do
        % extract first, second, third quartile pixel lengths
        $\mathrm{Q} = \mathrm{percentile}[\mathrm{PL}, \{\mathrm{Q1}, \mathrm{Q2}, \mathrm{Q3}\}]$
    end
    $\mathrm{L} = \mathrm{Q}[\{\mathrm{Q1}, \mathrm{Q2}, \mathrm{Q3}\}].\mathrm{mean}.\mathrm{max} + k * \mathrm{Q}[\{\mathrm{Q1}\}].\mathrm{std}$
    $\mathrm{M} = \mathrm{Q}[\{\mathrm{Q1}, \mathrm{Q2}, \mathrm{Q3}\}].\mathrm{mean}.\mathrm{max} + k * \mathrm{Q}[\{\mathrm{Q2}\}].\mathrm{std}$
    $\mathrm{H} = \mathrm{Q}[\{\mathrm{Q1}, \mathrm{Q2}, \mathrm{Q3}\}].\mathrm{mean}.\mathrm{max} + k * \mathrm{Q}[\{\mathrm{Q3}\}].\mathrm{std}$
end
Output: L, M, H
```

Figure 14. Adaptive Thresholding Steps

The queueing levels at a work zone, freeway and intersection is quantified using a Mask-RCNN framework. Figure 15 to 17 is used to demonstrate the results through heat map plots. Overall the model can visibly capture the start and dissipation of congestion events. The freeway and intersection were able to detect both AM and PM peak hours (See Figure 15 & 16). However, for a work-zone, only PM peak hour was detected (See Figure 17), and no AM peak hour was observed. Although, mostly, work-zone sites have active construction going on during the day-time. On further investigation, it was comprehended that work-zone activities at this site started after the AM peak hour, hence that resulted in lower levels of queueing during the AM hours.

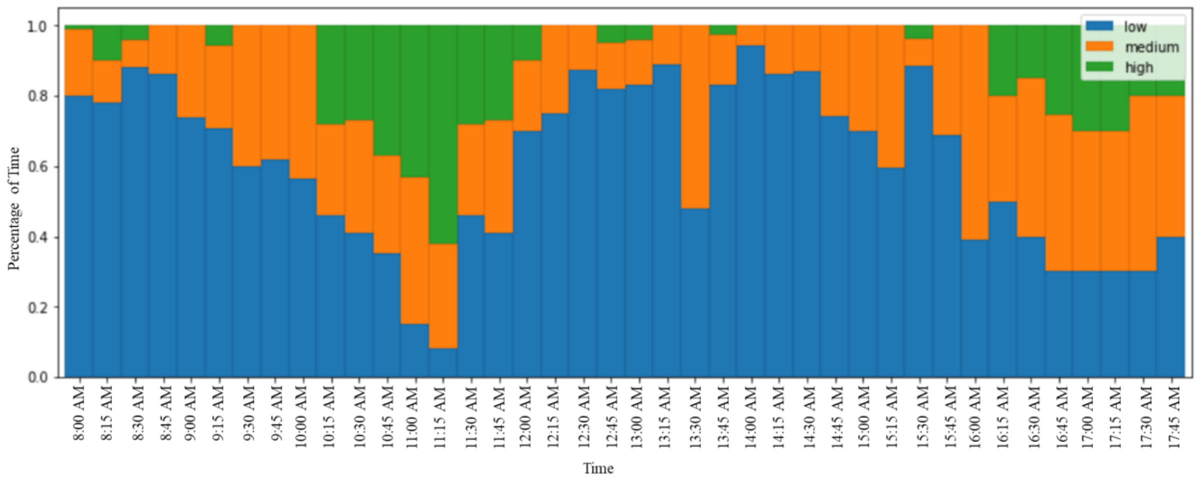


Figure 15. Heat map of traffic queue severity at freeway

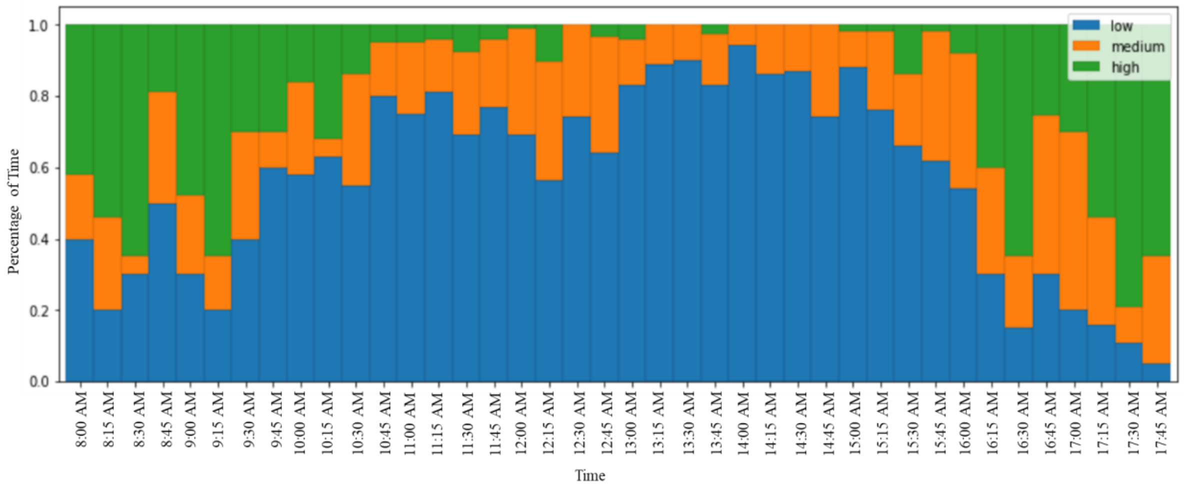


Figure 16. Heat map of traffic queue severity at intersection

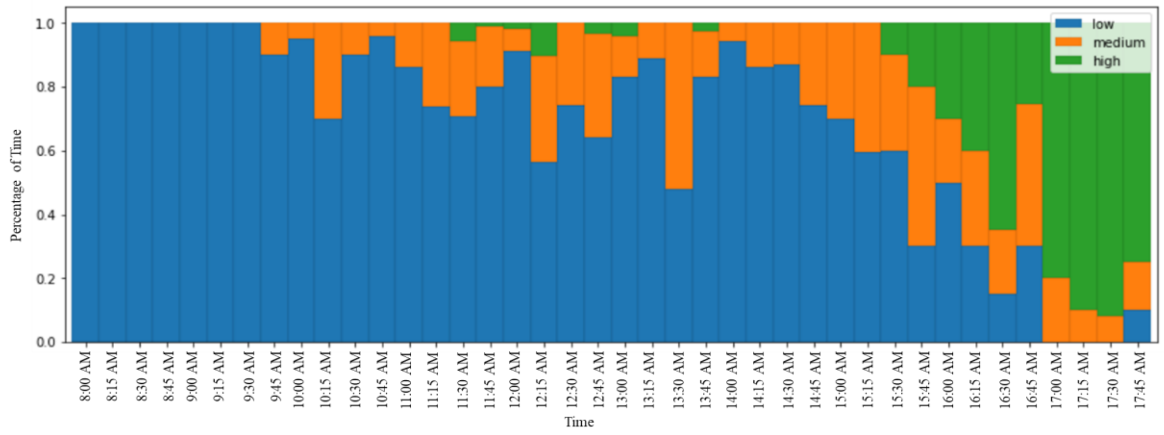


Figure 17. Heat map of traffic queue severity at work-zone

4.4.1.2 Bottlenecks and Challenges

Mask R-CNN takes approximately 0.3 seconds to process a traffic scene. A typical frame rate for CCTV cameras is 15 frames per second (fps). At this rate, the methodology developed in this paper cannot be used in real time. One way around this is to use YOLO (which can process 50 frames in one second) to process video feeds initially, if a scene is flagged as congested, Mask RCNN model can be called to extract the queue parameters for that particular scene. This way, the model is not running on every single frame from the traffic scene. Alternatively, feeds from CCTV cameras could be re-sampled at 1 fps instead of 15fps. Another bottleneck encountered was regarding how queues are described. A queue at an intersection could just be a platoon of vehicles on a freeway. Training the Mask R-CNN to be able to distinguish between queues at intersections and freeways was a challenge. Eventually, we had to create two different models: one for uninterrupted and the other for interrupted.

4.4.2 Traffic Anomaly Detection

Detecting vehicles effectively is the first step when it comes to finding out traffic anomalies on the roads. In this study, we used two deep learning-based object detection frameworks, YOLO and Faster R-CNN. The main purpose is to determine what model has a higher accuracy and faster processing speed and then perform a comparative analysis. Some of the sample detection results are shown in Figure 18.

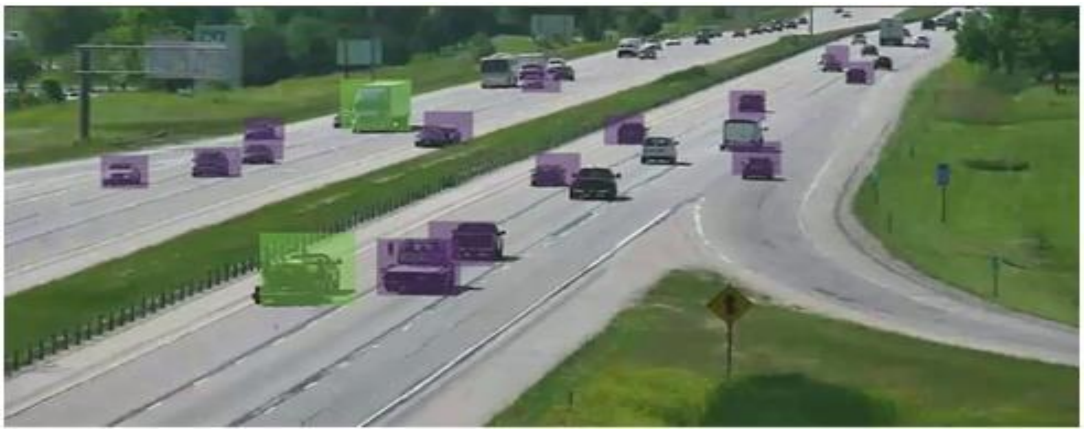
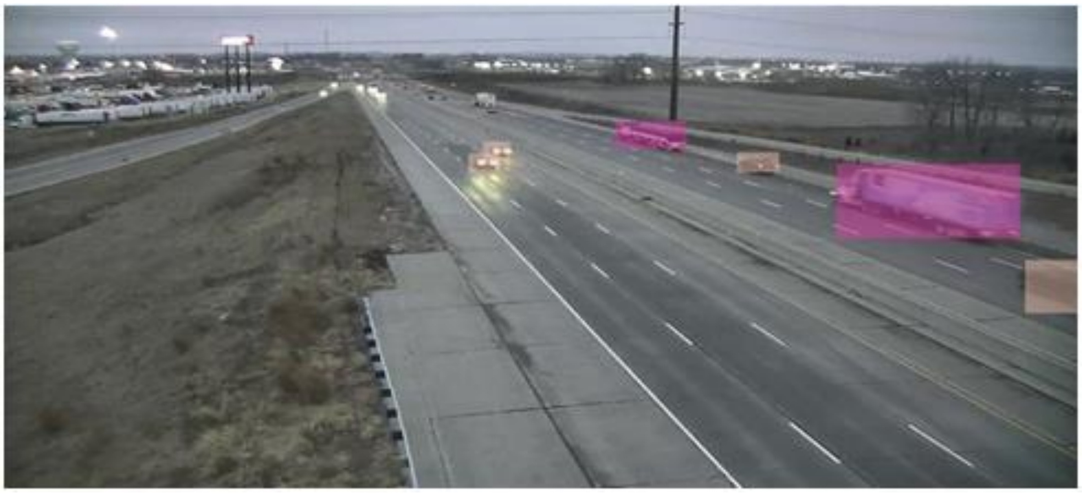



Figure 18. Vehicle detection results from YOLO

In order to understand and compare the results of both Faster-RCNN and YOLO frameworks, we used confusion matrices and F-1 scores obtained from both these models. The confusion matrix represents the level of accuracy for different sections of image classification. Altogether 25 results are shown in a 5*5 table which we refer to as a confusion matrix. Here, every row presents the actual number of predictions and total number of each row signifies the number of targets predicted for that class. Similarly, each column represents the true number of targets and the total number of each column signifies the actual number of targets for that class. Likewise, F-1 score as shown in equation (iv) is used to compare the performance of both Mask-RCNN and

YOLO models. The results obtained for confusion matrix and F-1 scores are shown in Tables 4-6.

$$F - 1 = \frac{2*Precision*Recall}{Precision+Recall} \qquad \text{(iv)}$$

YOLO					
Pred \ True	**Ped**	**Cyclist**	**Car**	**Bus**	**Truck**
Ped	0.992895204	0.005328597	0.000888099	0	0.000888099
Cyclist	0.02283105	0.97260274	0	0	0.00456621
Car	0	0	0.994734751	0.000198689	0.005066561
Bus	0	0	0	0.994285714	0.005714286
Truck	0	0	0.045793397	0.007454739	0.946751864
RCNN					
Pred \ True	**Ped**	**Cyclist**	**Car**	**Bus**	**Truck**
Ped	0.997370727	0.002629273	0	0	0
Cyclist	0.040178571	0.955357143	0.004464286	0	0
Car	0.00029163	0.00019442	0.994361816	0.00038884	0.004763293
Bus	0	0	0.010362694	0.979274611	0.010362694
Truck	0	0	0.0367428	0.007944389	0.95531281

Table 4. Confusion Matrices of Faster R- CNN and YOLO

YOLO			
Class	**Precision**	**Recall**	**F-1 score**
Ped	0.92165122	0.7367003	0.81886228
Cyclist	0.94247788	0.8658537	0.90254237
Car	0.92769723	0.7990594	0.85858674
Bus	0.95081967	0.8571429	0.9015544
Truck	0.91603875	0.840079	0.87641607
Total	0.92690348	0.7975653	0.85738408

Table 5. F-1 Scores of YOLO

Faster R-CNN			
Class	**Precision**	**Recall**	**F-1 score**
Ped	0.87549251	0.88385044	0.87965162
Cyclist	0.9380531	0.85140562	0.89263158
Car	0.83125773	0.87880373	0.85436976
Bus	0.8952381	0.86635945	0.88056206
Truck	0.89282203	0.8972332	0.89502218
Total	0.84178622	0.87989299	0.86041789

Table 6. F-1 Scores of Faster R- CNN

The results obtained for both Faster-RCNN and YOLO models in confusion matrices (See Table 4), show that the performance of both these models are alike. However, as seen from table 4, it appears that Faster-RCNN attained a weaker lead in detecting trucks, cyclists and pedestrians. Apart from these 3 classes, both Faster-RCNN and YOLO distinguished cars with 99% accuracy. As seen from the F-1 scores, YOLO seems to be more precise than Faster-RCNN. However, the recall values for Faster-RCNN is larger than the one obtained for YOLO meaning that the YOLO does not detect as many objects on a traffic scene as compared to Faster-RCNN. And with a total F-1 score of 0.86041789, it is evident that the Faster-RCNN was marginally better in comparison to YOLO (See Table 5 and 6 for more details).

Similarly, in order to excerpt higher level features, it is important to correctly detect objects from traffic feeds. After the detector detects the position of an object on a traffic scene, the tracker is accountable for generating the state of objects from a sequence of traffic frames (i.e. video). In this study, a multi-target tracking technique is used to initially associate detections to each exclusive roadway, and on the basis of these associations, the speed and state of each object is approximated. Both location-based and feature-based tracking methods were used for this study. Sections 4.4.2.1 and 4.4.2.2 explains both these multi-object tracking methods in more detail.

4.4.2.1 Tracking Detection by Intersection Over Union (IOU)

On calculating the spatial overlay for object detection boxes in each successive video frames, an IOU assigns detections. In this study, Eric et al.’s implementation was adopted [59]. Since IOU tracker has low computational cost, obtaining trajectories of vehicles could be achieved and be integrated to other high-order tracker without affecting the computed speed. A frame rate of over 50,000 frames/second could also be achieved with the use of IOU. It is important to note that IOU tracker is heavily reliant on how accurately recognitions are achieved by object detection models. To obtain higher performance of IOU, the researchers in [59] used visual tracking alongside IOU so that the number of ID switches and fragmentations can be reduced while keeping a higher order tracking speed [60]. As seen from Figure 19, the first image is classified a freeway and the second image is the intersection.

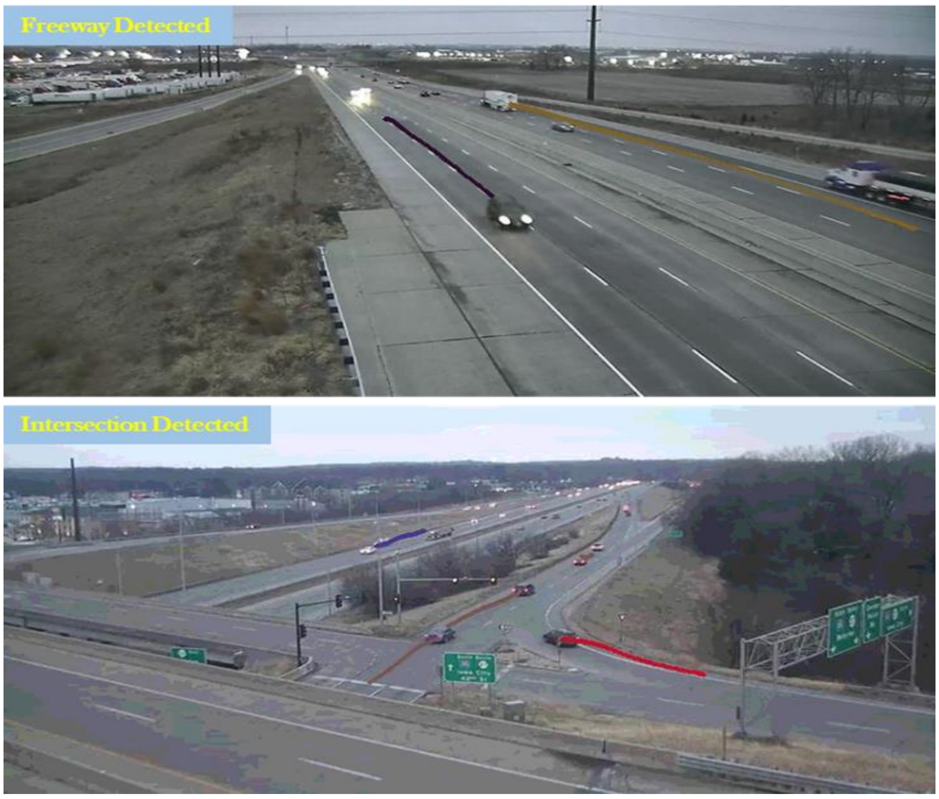


Figure 19. Vehicle tracking and road type classification

The travel path trajectories as obtained from IOU tracker is useful to classify the direction on which the vehicle travels which would ultimately help predict the kind of road that is, whether an intersection or a freeway. The direction of vehicle path trajectory is governed by the changes made to the x and y coordinates of the respective vehicle paths. Road type is classified based on the number of road directions detected.

In cases, where there are more than two detected directions, the road type is classified as either an intersection or an interchange. Similarly, for only two detected directions, the road would be classified as a freeway or a two-lane road. Some of the examples of trajectories and various road types are shown in figure 19.

4.4.2.2 Feature Tracker Based Object Tracking

In Feature-based object tracking system, there is the usage of appearance information to track objects in a respective traffic scene. This method is useful in tracking vehicles in occluded settings. The system extracts object features from one frame and then matches appearance information with successive frames based on the measure of similarity. The minimum value of cosine distance is useful at computing any resemblance between some of the characteristic features which is useful for vehicle tracking.

4.4.2.3 Results Comparison between IOU and Feature Tracker

Based on the average switch rate, a clustered column chart is used to compare detection results of IOU and Feature Tracker. The switch rate measures how frequently a vehicle is allotted a new track number when it crosses a traffic scene. In ideal terms, every vehicle should be assigned a single track and should not be counted multiple times. To track the performance of the tracker, each of them are tested on different set of conditions such as:

(i) Quality and content, good condition (fluent, clear video)

(ii) Occlusion

(iii) Manual control (whether zoomed in or out)

(iv) Weather conditions (snow, fog, etc.)

(v) Video pixilation and stuck conditions

(vi) Night time

Likewise, the switch rate is the ratio of vehicle switch to the actual number of vehicles.

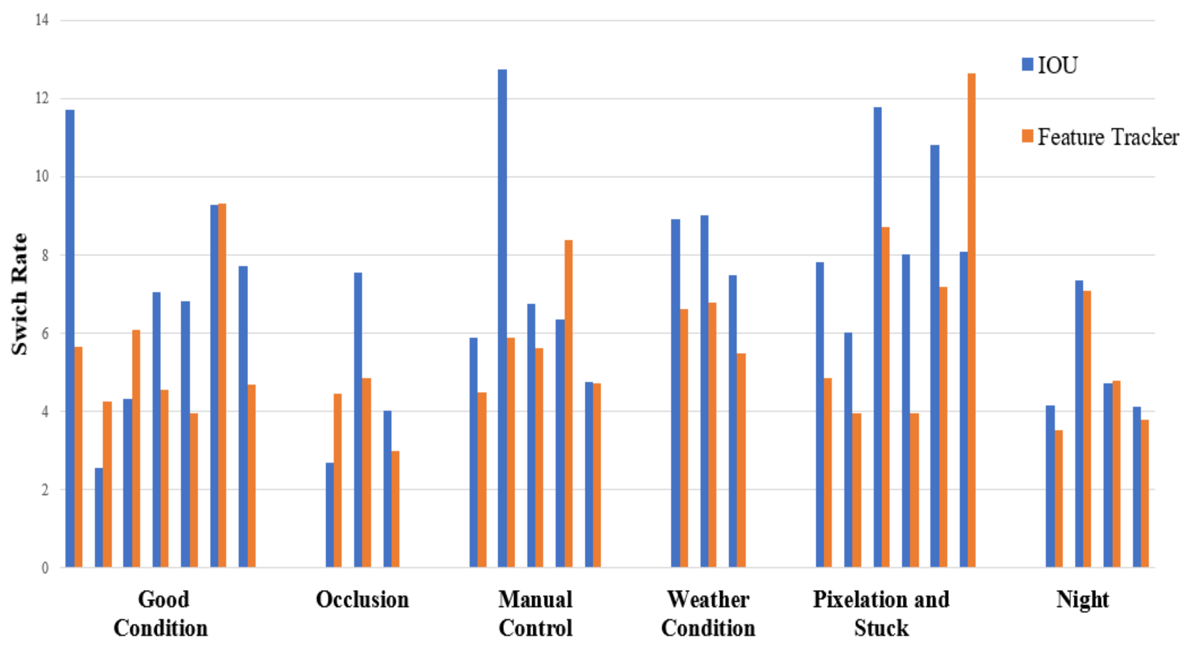


Figure 20. Comparison of clustered charts for IOU and Feature Tracker

The test results confirm that IOU tracker has a higher switch rate in comparison to the feature-based tracker. Pixilation and hostile weather conditions were some of the major factors that affected the accuracy of both these trackers. However, it is imperative to note that the feature tracker is highly robust in occluded environments. Similarly, the computational costs associated to feature tracker is relatively higher in contrast to the IOU tracker. Despite, the fact that IOU tracker underperformed the feature tracker, we used the IOU tracker for anomaly detection as it can process traffic video-feeds in real-time.

4.4.2.4 Anomaly Detection

In this study, an anomaly is defined as the one in which any vehicle stops for a total period of 15 seconds or more. For anomaly detection, the speeds of all tracked vehicles are tabulated over time. Based on that, any vehicle exceeding the speed of 0.5 pixels per second over a 15 second duration is labelled as a possible anomaly. Likewise, the direction of travel and the type of road is used to determine the likelihood of anomaly in post-processing steps. The detected traffic anomalies are shown in Figure 21. These anomalies are shown both prior to and after- post-processing of the required steps. The effects of ID switches from the IOU tracker is quite apparent in the second column of Figure 21. This, in fact causes multiple anomalies to be detected at the same position. In the post-processing step, an ID suppressing method is used to reduce the number of anomalies. For this, the first step is to detect multiple anomalies that remain close to each other and then they are merged into one. Then after, all the anomalies are merged based on the direction of the roadway. This assumes that no more than one anomaly exists on one side of the road under a 15-minute time duration. Finally, the traffic anomalies are plotted if a roadway is either a freeway or a two-lane road and discards the anomaly as false if the road is deemed as an intersection.

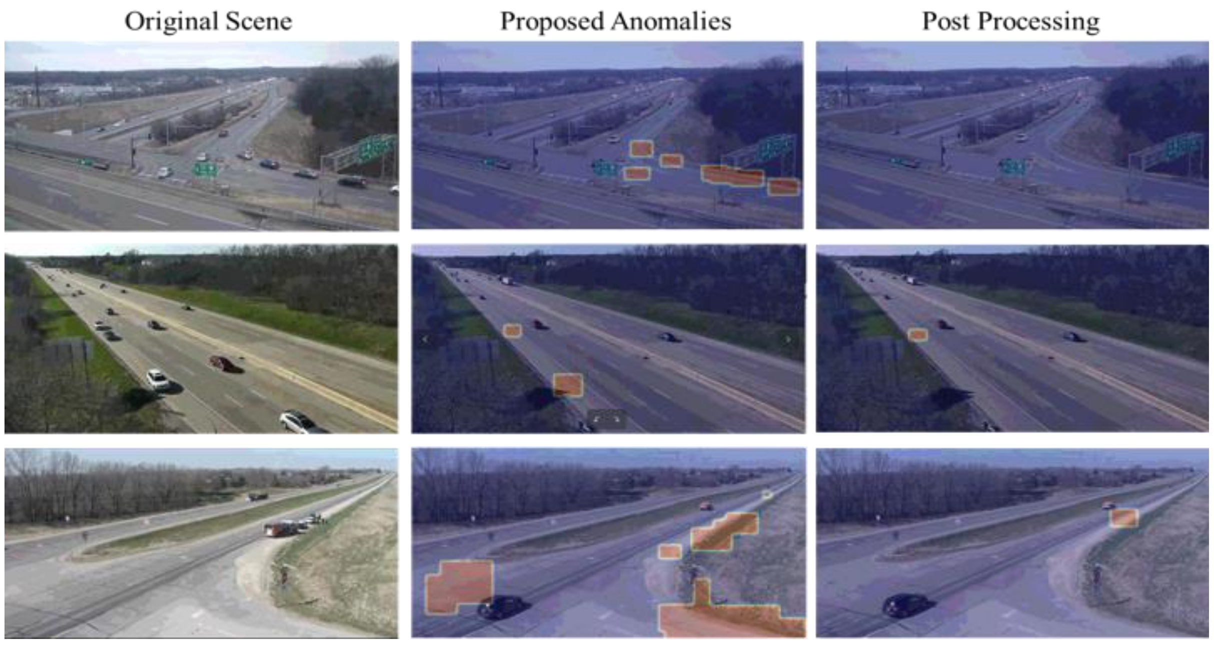


Figure 21. Traffic Anomalies

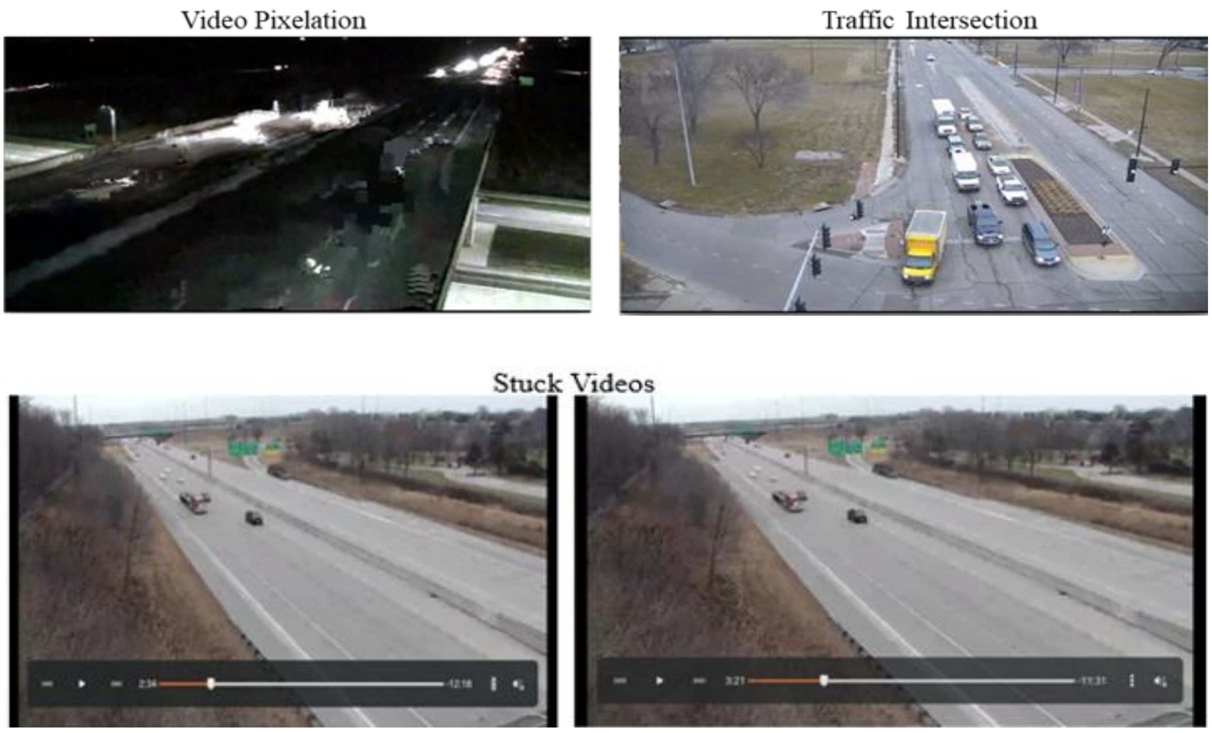


Figure 22. Challenges disturbing the performance of the proposed algorithm: Video Pixelation, Traffic Intersections, and Video Stuck for nearly 60 seconds.

The proposed traffic anomaly detection system was assessed on 100 traffic video-feeds with varied traffic and weather conditions [61]. The major challenge at detecting anomalies was largely due to the quality of traffic videos. Out of the various problems encountered, some of the most notable were:

(i) Video Pixelation

(ii) Videos Zoomed In/ Out

(iii) Videos Stuck for exceedingly larger periods

Video Pixelation: It is mostly caused due to the display of bitmap or a small portion of a bitmap at exceedingly larger sizes where individual pixels comprising of bit maps appear visible. It can also occur due to the compression of video files. Because of pixelation, the video files could create false detections over shorter durations of time which could be labelled as anomalies if not corrected timely. To lessen the drawbacks of video pixelation, the proposed algorithm discounted vehicles that remained still for less than 15 seconds time duration.

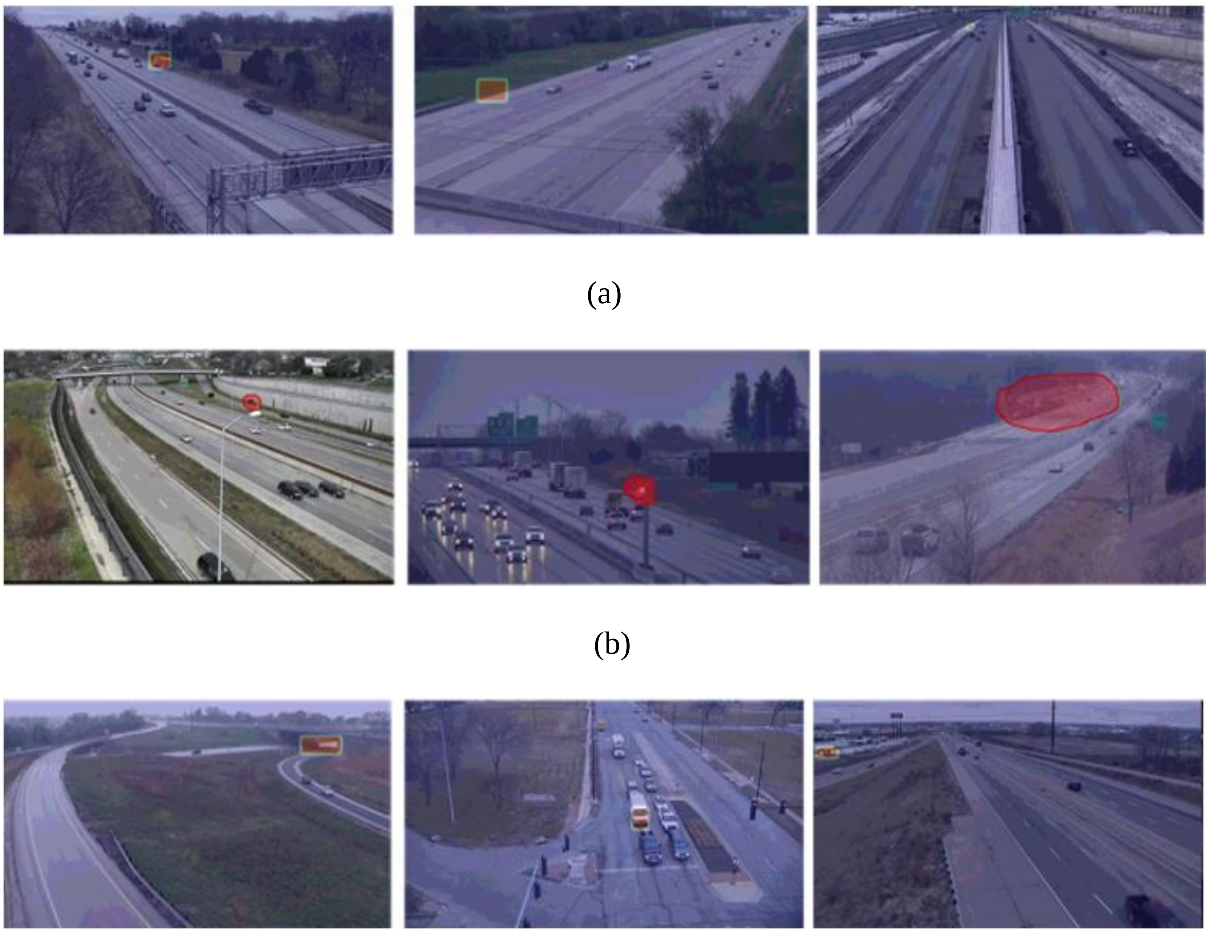
Intersection: In this study, an IOU tracker classifies a vehicle or a group of vehicles as a possible anomaly if a traffic stop sign requires a vehicle to stop. This condition could not be classified as an anomaly, but IOU classifies them as such. Therefore, in order to remedy this issue, one of the first steps is to determine whether a roadway is an intersection or a freeway. Based on the road type, any vehicle remaining stationary for more than 30 seconds on a freeway is considered an anomaly and for an intersection, the time limit is set at 60 seconds.

Stuck Video: Sometimes, the stuck video files challenges the anomaly detection capability of IOU tracker. Often, the videos remain stuck for over a minute. In such a case, the IOU tracker detects the vehicle as a possible anomaly every time the video gets stuck for exceedingly larger time periods. This is however, a false anomaly. The experiment found that although the video stuck for larger time periods, the speed of every vehicle in the stuck video remains 0, as it is the same video-frame scene. Since, the vehicle's speed involved in a crash is approximately 0 but not exactly zero, the rectangle surrounding it is in a slightly swaying state. Therefore, all anomalies with a speed of zero are classified as false detections.

To determine the performance of the proposed anomaly detection model, standard performance metrics of F1, Root Mean Square Error (RMSE) and S3 values were used. The equation used to compute the value of S3 score is shown in equation (1).

$$S3 = F1 * (1 - NRMSE) \qquad (1)$$

Here, NRMSE is the Normalized Root Mean Square Error. To compute the F-1 score, we require the value for True positive. A true positive is the one in which the detection of an anomaly is within the 10 seconds time frame from the actual. An anomaly can only be a true positive for a single anomaly. That is the same anomaly cannot be counted twice. False positives are the ones that do not resemble to true positives for certain occurrences. Similarly, false negatives are the anomalies that are in fact true anomalies but are missed by the model. Some of the examples of True Positives, False Positives and False Negative Results are shown in Figure 23.



(a)

(b)

(c)

Figure 23. Classification of predicted anomalies - First Row, (a): True Positives. Second Row, (b): False Negatives - anomalies indicated with red circles. Third Row, (c): False Positives

Errors in anomaly detection is represented by the root mean square error (RMSE). It is computed for the ground truth anomaly times and predicted anomaly times for any true positive's detections. To calculate S3 values, the RMSE is then normalized as shown by NRMSE in equation (1). For this study, normalization is done by performing a min-max normalization approach with the largest value being 300 and the lowest value being set to 0. As per the results shown in Table 7, the F1 score recorded is at a value of 0.8333, which means that the detector can detect approximately 83.3% of the overall anomalies. However, due to the limitations in the dataset, especially for vehicles located far away from the camera, the model failed to detect anomalies in those cases.

Name	**F1**	**RMSE**	**S3**
Model Score	0.8333	154.7741	0.4034

Table 7. F1, RMSE and S3 Final values

4.4.3 Vehicle Detection and Count

The goal here was to develop a one-look vehicle counting system that can automatically detect and compute the number of vehicles on the road. For our study, we tabulated the number of vehicles passing across the northbound and southbound roads and computed the accuracy of the model using the statistics precision, recall and F-1 values. An image showing the count of vehicles on both northbound and southbound is shown in Figure 24.

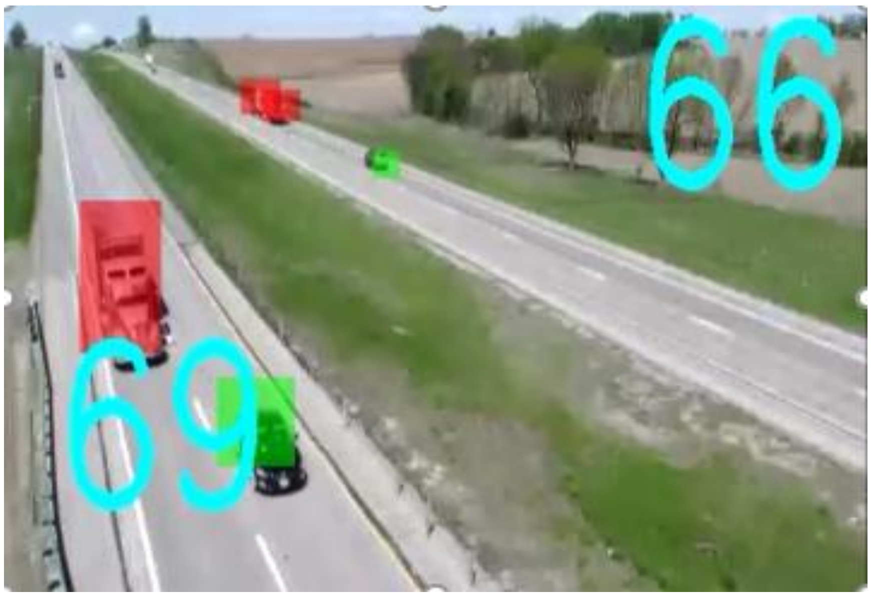


Figure 24. Automatic Vehicle Count across Northbound and Southbound directions

In order to evaluate the performance of the proposed model, we manually counted the number of vehicles passing across both northbound and southbound directions and computed the number of misses recorded by the proposed model. The total length of the traffic video was 15 minutes each and altogether 26 traffic videos were used for the sample study. There were traffic videos of all sort from day light, to night time and snow. The results obtained are shown is shown in Table 8. The accuracy of the proposed model is obtained by simply dividing the automatic vehicle count over the actual vehicle count (i.e. manual vehicle count). As seen from the table, the performance of the proposed model was much higher since these models obtained over 96% accuracies in both northbound and southbound directions.

Direction	Manual Vehicle Count	Automatic Vehicle Count	Accuracy
Northbound	13,796	14,251	96.81 %
Southbound	12,460	12,848	96.98 %

Table 8. Accuracy for Automatic Vehicle Count

CHAPTER 5 – CONCLUSIONS AND FUTURE RESEARCH

The rapid advancement in the field of machine learning and high-performance computing have highly augmented the scope of automatic traffic monitoring systems. In the current study, we implemented three deep learning-based algorithms, Mask-RCNN, Faster-RCNN and YOLO and deployed two different object tracking systems, IOU and Feature Tracker. Mask-RCNN was used to detect traffic queues from real-time video feeds whereas YOLO and Faster-RCNN were used for detecting traffic anomalies. Furthermore, YOLO was used for the comparison of test results obtained from Mask R-CNN and Faster R-CNN. To ensure uniformity, same dataset of images were used for testing purpose. For traffic queues prediction, Mask R-CNN achieved an accuracy of 92.8% while the highest accuracy achieved by YOLO was 95.5%. The inconsistencies in correctly detecting congestion was mostly due to the poor image quality, traffic queues located far away from the camera, single-lane blockages and glaring effects. All these issues significantly affected the accuracies of the proposed models. It is imperative to note that the performance capabilities of these models appeared to be higher during the day- time than at night. Likewise, for images with too many objects, queue detection wasn't very accurate which caused a small decline in the overall performance.

The training time for Mask-RCNN was relatively smaller which made it easier to train and implement on real-time video feeds. However, the biggest limitation of Mask-RCNN was that it required 0.3 seconds to process a traffic scene whereas a typical frame rate for a CCTV camera is around 15 fps. This could eventually cause lagging and make the post-processing of videos slower. On the other hand, YOLO is extremely fast, and has the ability of processing video feeds up to 50 frames per second. This is much faster than the actual CCTV video which stands at around 15 fps. Therefore, in this research we used YOLO to process video feeds initially, and if a scene is flagged as congested, Mask RCNN model is brought in to extract queue related parameters for that scene.

In order to track anomaly, IOU and Feature Tracker were used. For anomaly detection, the F1, RMSE and S3 scores for the model were found to be 0.8333, 154.7741, and 0.4034 respectively. From the results, it was quite evident that the model detected anomalies located closer to the camera but had issues with detecting distant anomalies. Factors such as video pixelation, and traffic intersections also contributed to the lower S3 scores. Video pixelation mainly occurred due to the data compression of video files. The pixelated videos created random false detections over short durations that led to anomalies which were not true. It is important to note that the tracker's performance was heavily reliant on the object detector. Similarly, videos stuck periodically caused the tracker to classify some of the false anomalies to a true positive, which was later corrected by studying the speed of the vehicles. The speed of vehicles in stuck video clip remained zero, since each frame were same; thus, the bounding box surrounding it was stationary. However, the speed of the vehicle involved in an accident could be close to zero but not equal to zero, as rectangle surrounding it is mostly in a swaying state. Hence, all anomalies with a speed of zero were classified as false anomalies, thereby discounting the misclassifications of the tracker. Likewise, some of the approaches such as anomaly suppression or video pixelation corrections were performed to improve the effectiveness of the proposed framework, thereby enabling an effective anomaly detection system. Similarly, the overall accuracy for the vehicle count was 96.81 % in the northbound and 96.98 % in the southbound direction. The accuracies were comparatively higher and could be implemented in real-time.

Future studies in this area could investigate a more robust traffic monitoring system using a larger image dataset and could use a combination of different model architectural designs to enhance the overall accuracies. In order to effectively study queueing parameters, the system could be further used to automatically calibrate different CCTV cameras that remain resolute to any changes in camera orientation so that it is able to accurately extract queue-length parameters in feet or meters. Similarly, the object detection algorithm could be further improved by adding additional information for instance, including vehicles and traffic scenes from different camera angles, black-and-white/coloured images in varied environmental conditions. Another future problem to investigate would be the occlusion where the vehicles are too close to each other. In addition, some other classifiers could also be considered for future study. Since, anomaly detection systems rely heavily on trackers and trackers rely more so on detectors, steps should be taken to improve the accuracy of the object detector and furthermore, improve tracking for improved post processing of anomalies using larger-real world datasets. The application of automatic vehicle counting could be further explored by improving the tracking algorithm and taking advantage of the automated vehicle counting system to studying its impact on effective traffic management system while reducing congestion bottlenecks on jammed intersections, freeways and work-zones.

REFERENCES

[1] Salvi, G., 2014, March. An automated nighttime vehicle counting and detection system for traffic surveillance. In *2014 International Conference on Computational Science and Computational Intelligence* (Vol. 1, pp. 131-136). IEEE.

[2] Wang, K., Li, Z., Yao, Q., Huang, W. and Wang, F.Y., 2007, December. An automated vehicle counting system for traffic surveillance. In *2007 IEEE International Conference on Vehicular Electronics and Safety* (pp. 1-6). IEEE.

[3] Chiu, C.C., Ku, M.Y. and Wang, C.Y., 2010. Automatic Traffic Surveillance System for Vision-Based Vehicle Recognition and Tracking. *J. Inf. Sci. Eng.*, *26*(2), pp.611-629.

[4] Coifman, B., Beymer, D., McLauchlan, P. and Malik, J., 1998. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C: Emerging Technologies*, *6*(4), pp.271-288.

[5] Willis, C., Harborne, D., Tomsett, R., & Alzantot, M. (2017). A deep convolutional network for traffic congestion classification. In Proc NATO IST-158/RSM-010 Specialists' Meeting on Content Based Real-Time Analytics of Multi-Media Streams (pp. 1-11).

[6] Chakraborty, P., Adu-Gyamfi, Y. O., Poddar, S., Ahsani, V., Sharma, A., & Sarkar, S. (2018). Traffic Congestion Detection from Camera Images using Deep Convolution Neural Networks. Transportation Research Record, 0361198118777631.

[7] Morris, T., Schwach, J. A., & Michalopoulos, P. G. (2011). Low-cost portable video-based queue detection for work-zone safety.

[8] Hao, C., & Yongyi, L. (2011). Research on Queue Detection Technology Based on Video for City Road Section. In ICTIS 2011: Multimodal Approach to Sustained Transportation System Development: Information, Technology, Implementation (pp. 652-661).

[9] Zhang, J.S., Cao, J. and Mao, B., 2017, July. Application of deep learning and unmanned aerial vehicle technology in traffic flow monitoring. In *2017 International Conference on Machine Learning and Cybernetics (ICMLC)* (Vol. 1, pp. 189-194). IEEE.

[10] Lin, C. P., Tai, J. C., & Song, K. T. (2003, September). Traffic monitoring based on real-time image tracking. In Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on (Vol. 2, pp. 2091-2096). IEEE.

[11] Zhou, J., Gao, D., & Zhang, D. (2007). Moving vehicle detection for automatic traffic monitoring. IEEE transactions on vehicular technology, 56(1), 51-59.

[12] Bas, E., Tekalp, A. M., & Salman, F. S. (2007, June). Automatic vehicle counting from video for traffic flow analysis. In *Intelligent Vehicles Symposium, 2007 IEEE* (pp. 392-397). IEEE.

[13] Zhu, Z., Xu, G., Yang, B., Shi, D., & Lin, X. (2000). VISATRAM: A real-time vision system for automatic traffic monitoring. Image and Vision Computing, 18(10), 781-794.

[14] Bautista, C. M., Dy, C. A., Mañalac, M. I., Orbe, R. A., & Cordel, M. (2016, May). Convolutional neural network for vehicle detection in low resolution traffic videos. In Region 10 Symposium (TENSYMP), 2016 IEEE (pp. 277-281). IEEE.

[15] Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., & Wang, Y. (2017). Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. Sensors, 17(4), 818.

[16] Yuan, Y., J. Wan, and Q. Wang. Congested scene classification via efficient unsupervised feature learning and density estimation. Pattern Recognition 56, 2016, pp. 159-169.

[17] McLauchlan, P., D. Beymer, B. Coifman, and J. Malik. A real-time computer vision system for measuring traffic parameters. In cvpr (p. 495). IEEE, 1997.

[18]https://hackernoon.com/deep-learning-vs-machine-learning-a-simple-explanation-47405b3eef08. Accessed 9/3/2019

[19]https://www.freecodecamp.org/news/want-to-know-how-deep-learning-works-heres-a-quick-guide-for-everyone-1aedeca88076/. Accessed 9/3/2019

[20] https://www.modot.org/facilities-optimization. Accessed 9/3/2019

[21] https://www.wsdot.wa.gov/Operations/Traffic/tmc.htm. Accessed 9/3/2019

[22] Ullman, G. L., V. Iragavarapu, and R.E. Brydia. Safety Effects of Portable End-of-Queue Warning System Deployments at Texas Work Zones. *Transportation Research Record: Journal of the Transportation Research Board*, No. 2555, 2016, pp. 46-52. https://doi.org/10.3141/2555-06.

[23] Fouladgar, M., Parchami, M., Elmasri, R., & Ghaderi, A. (2017, May). Scalable deep traffic flow neural networks for urban traffic congestion prediction. In *2017 International Joint Conference on Neural Networks (IJCNN)* (pp. 2251-2258). IEEE.

[24] Dougherty, M.S., Kirby, H.R., Boyle, R.D., "The use of neural networks to recognise and predict traffic congestion", Traffic engineering & control, vol. 34, no. 6, 1993

[25] Ma, X., Yu, H., Wang, Y., Wang, Y., "Large-scale transportation network congestion evolution prediction using deep learning theory", PloS one, vol. 10, no. 3, pp. e0119044, 2015.

[26] Wang, J., Gu, Q., Wu, J., Liu, G., & Xiong, Z. (2016, December). Traffic speed prediction and congestion source exploration: A deep learning method. In 2016 IEEE 16th International Conference on Data Mining (ICDM) (pp. 499-508). IEEE.

[27] Skabardonis, A., and N. Geroliminis. Real-time Monitoring and Control on Signalized Arterials. Journal of Intelligent Transportation Systems, 2008, 12 (2), pp. 64-74. https://doi.org/10.1080/15472450802023337.

[28] Bezuidenhout, J.J., P. Ranjitkar, and R. Dunn. Estimating Queue at Traffic Signals. The Open Transportation Journal, 2014, 8, pp. 73-82.

[29] Hourdos, J. Development of a Queue Warning System Utilizing ATM Infrastructure System Development and Field Testing. Minnesota Department of Transportation, 2017, available at: http://mndot.gov/research/reports/2017/201720.pdf.

[30] Dinh, T.U.J., R. Billot, E. Pillet, and N.E.E. Faouzi. Real-Time Queue-End Detection on Freeways with Floating-Car Data: Practice-Ready Algorithm. Transportation Research Record: Journal of Transportation Research Board, No. 2470, 2014, pp. 46-56. https://doi.org/10.3141/2470-05.

[31] Wang, Z., Q. Cai, B. Wu, L. Zheng, and Y. Wang. Shockwave-Based Queue Estimation Approach for Under-saturated and Over-saturated Signalized Intersections Using Multi-Source Detection Data. *Journal of Intelligent Transportation Systems*, Vol. 21, Issue 3, 2017, pp. 167-178.

[32] Adu-Gyamfi, Y.O., A. Sharma. Comprehensive Data-Driven Evaluation of Wide-Area Probe Data: Opportunities and Challenges. Presented at 95th Annual Meeting of the Transportation Research Board, Washington, D.C., 2016

[33] Cheng, Y., X. Qin, J. Jin, and B. Ran. An Exploratory Shockwave Approach to Estimating Queue Length Using Probe Trajectories. *Journal of intelligent transportation systems*, Volume 16, Issue 1, 2012, pp. 12-23.

[34] Mundhenk, T. N., Konjevod, G., Sakla, W. A., & Boakye, K. (2016, October). A large contextual dataset for classification, detection and counting of cars with deep learning. In European Conference on Computer Vision (pp. 785-800). Springer, Cham.

[35] Moranduzzo, T., Melgani, F.: Automatic car counting method for unmanned aerial vehicle images. IEEE Transactions on Geoscience and Remote Sensing 52(3) (2014) 1635–1647

[36] Kamenetsky, D., Sherrah, J.: Aerial car detection and urban understanding. In: IEEE Conference on Digital Image Computing: Techniques and Applications (DICTA). (2015)

[37] Kamenetsky, D., Sherrah, J.: Aerial car detection and urban understanding. In: IEEE Conference on Digital Image Computing: Techniques and Applications (DICTA). (2015)

[38] Arteta, C., Lempitsky, V., Noble, J.A., Zisserman, A.: Interactive object counting. In: ECCV. (2014)

[39] Zhuang, P., Shang, Y., & Hua, B. (2009). Statistical methods to estimate vehicle count using traffic cameras. *Multidimensional Systems and Signal Processing*, *20*(2), 121-133.

[40] Kamijo, S., Matsushita, Y., Ikeuchi, K., & Sakauchi, M. (2000). Traffic monitoring and accident detection at intersections. *IEEE transactions on Intelligent transportation systems*, *1*(2), 108-118.

[41] Gangisetty, R., "Advanced traffic management system on I-476 in Pennsylvania," in Proc. IEEE ITSConf '97

[42] Rojas, J. C., and Crisman, J.D., "Vehicle detection in color images," in Proc. IEEE ITS Conf '97

[43] Zeng, N., and Crisman, J.D., "Vehicle matiching using color," in Proc. IEEE ITS Conf '97

[44] Lai, A. H. S., and Yung, N. H. C., "A video-based system methodology for detecting red light runners," in Proc. IAPR Workshop on MVA '98, pp. 23–26.

[45] Thajchayapong, S., Garcia-Trevino, E. S., & Barria, J. A. (2012). Distributed classification of traffic anomalies using microscopic traffic variables. *IEEE Transactions on Intelligent Transportation Systems*, *14*(1), 448-458.

[46] Ikeda, H., Kaneko, Y., Matsuo, T., & Tsuji, K. (1999, October). Abnormal incident detection system employing image processing technology. In *Proceedings 199 IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems (Cat. No. 99TH8383)* (pp. 748-752). IEEE.

[47] Michalopoulos, P., Jacobson, R., "Field Implementation and Testing of Machine Vision Based Incident Detection System," In Transportation Research Record: Journal of the Transportation Research Board, No. 1394, TRB, National Research Council, Washington, D.C., pp. 1-7, 1993.

[48] Ki, Y. K., Kim, J. W., & Baik, D. K. (2006, August). A traffic accident detection model using metadata registry. In Fourth International Conference on Software Engineering Research, Management and Applications (SERA'06) (pp. 255-259). IEEE.

[49] Oh, C., Oh, J. -S., and Ritchie, S., "Real-time hazardous traffic condition warning system: framework and evaluation," Intelligent Transportation Systems, IEEE Transactions on, vol. 6, no. 3, pp. 265–272, Sept 2005.

[50] Lee, C., Hellinga, B., and Saccomanno, F., "Real-time crash prediction model for the application to crash prevention in freeway traffic," in Proc. Transp. Res. Board Nat. Acad., Washington, D.C., 2003, pp. 68–77.

[51] Bochinski, E., Senst, T., & Sikora, T. (2018, November). Extending IOU based multi-object tracking by visual information. In 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (pp. 1-6). IEEE.

[52] Dogru, N., & Subasi, A. (2012, May). Traffic accident detection by using machine learning methods. In Third International Symposium on Sustainable Development (ISSD'12) (p. 467).

[53] Yun, K., Jeong, H., Yi, K. M., Kim, S. W., & Choi, J. Y. (2014, August). Motion interaction field for accident detection in traffic surveillance video. In 2014 22nd International Conference on Pattern Recognition (pp. 3062-3067). IEEE.

[54] Dutta, A., & Zisserman, A. (2019). The {VIA} Annotation Software for Images, Audio and Video. arXiv preprint arXiv:1904.10699, 5.

[55] https://github.com/Cartucho/OpenLabeling

[56] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 2961-2969).

[57] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

[58] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91-99).

[59] Bochinski, E., V. Eiselein, and T. Sikora. High-Speed tracking-by-detection without using image information. 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2017, pp. 1-6.

[60] Bochinski, E., T. Senst, and T. Sikora. Extending IOU Based Multi-Object Tracking by Visual Information. 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2018, pp. 1-6.

[61] The AI City Challenge. https://www.aicitychallenge.org/. Accessed April 10, 2019

[62] Wei, J., Zhao, J., Zhao, Y., & Zhao, Z. (2018). Unsupervised anomaly detection for traffic surveillance based on background modeling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 129-136).

[63] Xiaobin Liu, Shiliang Zhang, Qingming Huang, and Wen Gao. Ram: a region-aware deep model for vehicle reidentification. In 2018 IEEE International Conference on Multimedia and Expo (ICME), pages 1–6. IEEE, 2018.

[64] T. Schlegl, P. Seebock, S. M. Waldstein, U. Schmidt-Erfurth, ¨ and G. Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In International Conference on Information Processing in Medical Imaging, pages 146–157, 2017.

[65] Jin, P., Mandal, V., Shu, X., Adu-Gyamfi, Y., "Detecting Traffic Anomalies Using a Vision Based System", 99th Annual Meeting of the Transportation Research Board/Transportation Research Record, 2020.

[66]https://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/. Accessed on 11/10/2019

[67] Mandal, V., Uong, L. P., Jin, P., & Adu-Gyamfi, Y. O. (2019). Traffic Queue Monitoring with Mask Region-Based Convolutional Neural Network (No. 19-00850).